OS/2 Only!

Das fortlaufende Sammelwerk nur für OS/2

Band 2, Ausgabe 2/99, 1. Jahrgang

- **Neuigkeiten:**
  Treiber- und Softwareübersicht, Teil 1
- **Hardware:**
  OS/2-taugliche CD-ROM-Recorder
- **Software:**
  RSJ CD-Writer für OS/2
- **Konfigurieren und Anwenden:**
  Wartungspartitionen erstellen
- **Programmierung:**
  Workshop - DLLs, Teil 2
- **Netzwerk:**
  HTML-Workshop, Teil 2
  TCP/IP Grundlagen
- **Mutimedia:**
  Raytracing mit POV-Ray, Teil 1

CEF C.-E. Fischer
Buchverlag, Stuttgart

OS/2 Only!

Das fortlaufende Sammelwerk für OS/2

Bd. 2, Ausgabe Nr. 2/99
1. Jahrgang

Impressum

Die *OS/2 Only!* ist ein unabhängiges, endanwenderorientiertes OS/2-Magazin und wird herausgegeben vom C.-E. Fischer Buchverlag, Stuttgart.

Erscheinungsweise: Zweimonatlich zur Monatsmitte, 6 Ausgaben pro Jahr.
Anzeigen- und Redaktionschluß: Jeweils der 25. eines Monats vor dem Erscheinungsmonat.

Kontaktmöglichkeiten:
C.-E. Fischer Buchverlag
Bereich: OS/2 Only!
Wegländerstr. 24
D-70563 Stuttgart (Vaihingen)
Tel.: 0711-7802260 (Mo.-Fr. Von 9.30 bis 18.30 Uhr)
Fax: 0711-7802261
eMail: cefischer@t-online.de
WWW: http://www.cefischer.de

Abonnementsverkauf und Anzeigenaufnahme: Thomas Ropielewski
Lektorat: Carsten Momberg
Karikaturen: Ina Becker
Verantwortlich für den Inhalt: Clemens-Elias Fischer
Reprographie: Schwabenrepro GmbH, Stuttgart
Druck u. Verarbeitung: Druckerei Jauch GmbH, Stuttgart

1. Aufl., 1999
Printed in Germany

ISSN 1436-8307

Inhalt

Hinweise zum Gebrauch

Im folgenden Abschnitt finden Sie Hinweise zum Gebrauch unseres Sammelwerkes. Arbeiten Sie mit der *OS/2 Only!* zum ersten Mal, sollten Sie diesen Abschnitt aufmerksam durchlesen. Sind Sie mit dem Gebrauch des Magazins vertraut, möchten wir Ihnen empfehlen, immer mal wieder einen Blick auf diese Hinweise zu werfen, da sich von Zeit zu Zeit etwas ändern kann.

Grundsätzlicher Aufbau
Die *OS/2 Only!* gliedert sich in der Regel in folgende Kapitel, gegebenenfalls mit umfangreicheren, eigenen Abschnitten:

- Editorial
- Neuigkeiten

  Mit einem Schwerpunktthema und den Abschnitten »Kurzmeldungen, Bezugspunkte und Kontakte« sowie »Leserbriefe«, wenn diese dem Verlag vor Redaktionsschluß vorliegen. Da innerhalb des Zeitraums zur Vorbereitung einer neuen Ausgabe mitunter nicht genügend neue Links oder Kontaktadressen vorliegen, können Listen mit Links in der ein oder anderen Ausgabe auch ausgelassen werden.
- Hardware

  Mit den ständigen Abschnitten Theorie und Praxis, die meist eine thematisch zusammenhängende Einheit bilden. Außerdem (jedoch i.d.R. nur in jeder 2. Ausgabe): Eine umfassende Liste mit Fundorten für Treiber und eine Listung OS/2-tauglicher Hardware. Die Listen werden in jeder 2. Ausgabe erweitert bzw. korigiert.
- Software

  Mit einem Schwerpunktthema und den Abschnitten Share- und Freeware sowie einem Anwendungsworkshop. Außerdem (jedoch i.d.R. nur in jeder 2. Ausgabe): Eine umfassende Liste mit nativer OS/2-Software, deren Hersteller und Händler sowie einer kurzen Beschreibung zu jedem Produkt. Die Listen werden in jeder 2. Ausgabe erweitert bzw. korrigiert.
- Know how

  Mit einem Schwerpunktthema und den Abschnitten: Tips & Tricks, OS/2-Basiswissen.
- Programmieren

  Mit einem Schwerpunktthema oder einem Workshop und den Abschnitten: Java, C/ C++, Pascal und REXX.
- Multimedia

  Mit einem Schwerpunktthema und den Abschnitten: Hardware, Software.
- Netzwerk

  Mit einem Schwerpunktthema in Theorie und / oder Praxis und den Abschnitten: DFÜ, Internet, TCP/IP.
- Großkunden

  Mit den Abschnitten: Firmenprofil, Produkte. Diese Rubrik ist nicht in jedem Band enthalten.
- In eigener Sache

  Mit den Abschnitten: Druckwerke, Software, Besondere Angebote.
- Anhang, Stichwortverzeichnis, Lexikon.

Von Ausgabe zu Ausgabe können die Abschnitte einzelner Kapitel in ihrem Umfang variieren oder auch ganz ausbleiben. Das gleiche gilt für das Kapitel *Multimedia*, das zugunsten anderer Abschnitte bei Mangel an Informationen oder Artikeln ausgelassen wird.

Satz und Typographie
In der Kopfzeile einer jeden Seite finden Sie das jeweilige Kapitel und dessen Abschnitt wieder, in dem Sie sich gerade befinden. Kapitelname und Bezeichnung des Abschnitts sind mit einem Doppelpunkt voneinander getrennt. Alle Abschnitte sind im Inhaltsverzeichnis wiedergegeben.

Tabellen und/ oder Abbildungen sind jeweils innerhalb eines Artikels numeriert. Eine Auflistung aller Abbildungen und Tabellen eines Bandes finden Sie im Abbildungsverzeichnis am Ende eines jeden Bandes nach der Abbildungsnummer geordnet.

Für das Suchen nach bestimmten Themen können Sie auch das Stichwortverzeichnis benutzen, das nach Stichworten alphabetisch geordnet angelegt ist.

Artikel beginnen in einem Kapitel mit einer **fettgedruckten** Überschrift, die etwas größer als der normale Fließtext ist. Einzelne Abschnitte eines Artikels sind vom jeweils vorangegangenen Abschnitt mit einer Leerzeile getrennt und werden mit einer *kursiven* oder einer ***fettkursiven*** Überschrift eingeleitet.

Im Text selbst finden Sie Produktnamen, Herstellerbezeichnungen usw., die gerade angesprochen werden, *kursiv* gedruckt, also etwa: Das *ZIP-100* von *Iomega*.

Bestimmte Bezeichnungen, vor allem technische Ausdrücke, können **fett und unterstrichen** dargestellt sein. In einem solchen Fall können Sie den auf diese Weise hervorgehobenen Begriff im Lexikon nachschlagen, das Sie jeweils am Ende eines Bandes finden. Die einzelnen in einer Ausgabe gekennzeichneten Begriffe sind im lexikalischen Teil alphabetisch geordnet. Im Lexikon aufgenommene Begriffe können auch Verweise auf andere Bände tragen, sofern Sie in deren lexikalischen Teil bereits erklärt worden sind. Eine Zusammenstellung der lexikalischen Teile aller Bände finden Sie im Index-Band zu jedem Jahrgang.

❑ Aufzählungen werden mit dem Symbol ❑ eingeleitet. Nummerierungen erfolgen mit arabischen Ziffern. In Nummerierungen eingebettete Aufzählungen werden mit einem o kenntlich gemacht.

! *Wichtige Abschnitte, etwa besondere Hinweise, werden durchgängig fett und kursiv gedruckt und mit einem Ausrufezeichen versehen.*

Einzelne Schritte einer Vorgehensweise werden mit den Zahlensymbolen ❶❷❸❹❺❻ usw. geordnet und kenntlich gemacht.

Tip: *Ein besonderer Tip zur Konfiguration bzw. Handhabung eines bestimmten Programms oder einer Hardwarekomponente, werden mit einem kursiv gedrucken, großen Tip hervorgehoben. Der folgende Text wird außerdem kursiv wiedergegeben.*

Solutions for a Small Planet

Ja, ja, wir wissen auch, was IBM damit meint: Durch Netzwerke wird die Welt zum Dorf. Aber wir fanden noch eine andere Bedeutung für den gar nicht so unpassenden wenn auch etwas übertriebenen Werbespruch, an die IBM sicher nicht gedacht hat: Daß die OS/2-Gemeinde einen kleinen Wandelstern darstellt, der um einen großen blauen Riesen langsam seine Kreise zieht. Ich begrüße Sie zur Lektüre des 2. Bandes der *OS/2 Only!*

Wir fanden diese Auslegung so treffend, daß wir in unserem POV-Ray-Workshop das Ganze gleich in ein Bild umsetzen mußten, das bekanntlich mehr als tausend Worte sagt. Die Graphik aus dem Reich der virtuellen Realität wird Ihnen bestimmt gefallen, aber ich möchte nichts vorwegnehmen - alles zu seiner Zeit. Bevor Sie in das Multimedia-Kapitel gelangen, erwarten Sie Beiträge rund um die CD-ROM, wie sie aufgebaut ist, wie man sie herstellt, und was Sie dazu brauchen. Außerdem viele Neuigkeiten, Berichte zu den aktuellen Fixpacks für Warp 3 und 4 und natürlich viele andere nützliche Tips & Tricks rund um »unser« Betriebssystem. Viele Ihrer Verbesserungsvorschläge haben wir bereits in diesem Band in die Realität umgesetzt, angefangen vom Erscheinungsbild und der Papierqualität bis hin zum Inhalt. Sicher werden Ihnen die Veränderungen auffallen und -so hoffe ich- auch gefallen. Eine der tiefgreifensten Änderungen erfährt unsere Rubrik *Programmieren*, und darauf möchte ich an dieser Stelle noch einmal zu sprechen kommen.

Die Haltung so mancher Menschen ist schon merkwürdig. Da ist jemand unzufrieden -womit auch immer-, kennt den Grund jener Unzufriedenheit aber räumt damit nicht auf. So langsam aber sicher könnte man zu der Überzeugung kommen, es ist chic, unzufrieden zu sein. Wie es sich mit dem Privatleben gewisser Zeitgenossen verhält, so sieht es anscheinend auch in Sachen Computern aus: *»Computer sind kompliziert, man versteht sie nicht, der Umgang mit ihnen führt ins Chaos. Das war schon immer so, das ist so und wird auch so bleiben. Und wenn nicht, dann stimmt es nicht!«*

Auf diesen Schluß zu kommen, ist kein Problem, wenn man die vielen Windows-Benutzer bedenkt, für die das Wort *Computer* ein Synonym für *Chaos* ist. Aber woher kommt das? Unwissenheit. Oder: Vom Unwillen, das zugegebenermaßen komplexe Thema *Computer* verstehen zu wollen. Anwenden alleine ist es eben nicht. Niemand fährt schließlich Auto, nur weil er weiß, wo Bremse und Gaspedal sind. Dazu gehört mehr. Man sollte also schon Bescheid wissen - um Hardware, Software und natürlich das Betriebssystem.

Die Frage jedoch ist, wie z.B. eine hilflose Benutzerin ein Betriebssystem auswählen soll, wenn sie gar nicht weiß, was das ist. Da wurde ich vor einiger Zeit gefragt: *»Ist denn die STRG-Taste im Lieferumfang der Tastatur enthalten, oder muß ich die dazukaufen?«*- oder: *»Ich brauche einen Modemaccount? Ja, ist das denn überhaupt legal?«*- oder: *»Kopieren? Verschieben? Löschen? Datei? Kopieren kenne ich. Verschieben nicht. Löschen, da muß etwas weg sein, oder? Aber - was ist eine Datei?«* Mal ehrlich: So einem Menschen können Sie alles mögliche von Computern, Verzeichnissen, Laufwerken und Betriebssystemen erzählen: Er wird es glauben. Und der Markt erzählt leider viel.

So wird es mit der Zeit müßig, immer wieder zu erklären, was ein Verzeichnis,

eine Datei, ein Laufwerk und erst recht was ein Betriebssystem ist. Die Trennung zwischen den Begriffen *Windows* und *Computer* aufzuheben, ist kaum möglich und scheitert am Unwillen all derer, die nicht lernen wollen: Hauptsache die Kiste läuft, und ich kann damit schreiben! So, das war's. Was darauf läuft, können Sie im wesentlichen nur durch Schönreden beeinflussen: Ja, nimm das, das ist besser, nimm nicht dies oder jenes, das ist schlechter...- aber mehr kann man nicht tun.

Anders ist das bei all jenen, die verständiger sind oder es werden wollen. Wenn die Bereitschaft vorhanden ist, seine Kenntnisse zu vertiefen, dann fällt jeder und jedem schnell auf, daß einige Änderungen in puncto Betriebssystem am eigenen System anstehen. Nicht selten wird dann auch mal OS/2 ausprobiert, und als wir hörten, daß auch Anfänger und eine ganze Anzahl Umsteiger mit der *OS/2 Only!* viel anfangen konnten, waren wir erfreut und ermuntert, das Konzept weiter auszubauen, um wichtige Bereiche rund um OS/2 zu fördern und Anfängern tiefere Einblicke in das Reich der Datenverarbeitung zu geben; Berührungsängste abzubauen und den Schritt von anderen Systemen zu OS/2 zu wagen. Eine nicht ganz einfache Aufgabe.

Wir sind der Ansicht, daß mit der Verfügbarkeit v.a. kleiner Anwendungen, die im Privatbereich von großer Bedeutung sind (und wenn es nur um das Designen von Visitenkarten geht), OS/2 attraktiver wird. Derartige Programme können auch von weniger erfahrenen Programmierern entwickelt werden. Also werden wir die Anwendungsentwicklung mit OS/2 auf allen Ebenen fördern und schon ab Band 3 mit unseren Programmierreihen beginnen, die auch dem Anfänger, der noch nie programmiert hat, den Weg in die OS/2-Programmierung ebnen. Mit einer ausführlichen Reihe zur *REXX-Programmierung* geht es los. Ab Bd. 4 folgt eine gleiche Reihe für *Java*; und ab Bd. 7 die beiden Reihen für *C* und *Pascal*. Schritt für Schritt werden wir in die Programmierung einführen und zwar streng OS/2-spezifisch. Nach den einführenden Beiträgen werden wir uns auch komplexen Problemen aus der Informatik widmen, so daß sicher für alle etwas Interessantes dabei sein wird. Sprachreferenzen werden ebenso fester Bestandteil sein wie Beispielprogramme, Entwicklungswerkzeuge und viele Tips, die beim Programmieren in den genannten Sprachen helfen. Wer Interesse an der OS/2-Programmierung hat, wird mit der *OS/2 Only!* die richtige Begleitung finden - egal ob Anfänger oder Profi. Das Interesse an einer Windowsalternative ist vorhanden, und dem Computer-Dummy ist es wie gesagt egal, was er verwendet - es muß nur seine Bedürfnisse befriedigen.

So hoffen wir mit den vielen Verbesserungen und neuen Artikelreihen, die Sie in den folgenden Bänden finden werden, so manchem Computeranwender den Weg zu OS/2 zu ebnen und Unzufriedenheiten abzubauen, auch wenn weiterhin viele um das MS-Fensterprogramm ihre Kreise ziehen und ziehen werden. Denn über diejenigen, die des Lernens nicht fähig oder willens sind, sollte man sich nie ärgern (auch wenn es nicht um Computer geht). Man denke dann einfach an Kurt Tucholski:

»Wer genug von dieser Welt gesehen hat, lächelt, legt die Hände auf den Bauch.

Und schweigt.«-

Ihr

Clemens-Elias Fischer

Neue Fixpacks fürs System

Für Warp gibt es neue Fixpacks, nämlich das FP 38 für Warp 3 und das FP 9 für Warp 4. Mittlerweile sind auch schon die FPs 40 für Warp 3 und 10 für Warp 4 verfügbar. Sie enthalten gegenüber ihren Vorgängern keine wesentlichen Änderungen; jedoch sollte man sein Warp 3 mit dem FP 38, bzw. ein Warp 4 mit dem FP 9 fixen. Wie Sie das am besten machen, haben wir im *Know how* Kapitel im Artikel *Alles über Fixpacks* ausführlich erklärt.

Warum braucht man's?
Mit einem ungefixten Warp kommt man in bestimmten Situationen manchmal nicht weiter. So sind nicht nur neue Funktionen in den beiden Fixpacks enthalten, sondern auch einige neue Dateien, die einem das Leben leichter machen. Daneben eine verbesserte Hardwareunterstützung und einige weitere wichtige Dinge, die man haben muß bzw. haben sollte.

Die beiden Fixpacks können Sie übrigens bedenkenlos installieren (was leider nicht für jedes Fixpack gilt). Wir haben die Installation mehrmals auf verschiedenen Rechnern durchgeführt und konnten die Systeme nach den Korrekturen ohne Schwierigkeiten weiterhin betreiben. Im übrigen, um Fragen vorzubeugen, sind die Fixpacks kumulativ, d.h. sie enthalten alle vorangegangenen Korrekturen, so daß die neuen FPs direkt installiert werden können, ohne daß zuvor ein jüngeres Fix aufgespielt wurde.

Das Problem 2000
Über die Problematik mit dem Jahr 2000 und Computersystemen wurde schon in der Presse zur Genüge berichtet. Als unabwendbares Computerchaos wurde und wird das Ganze dargestellt. So unabwendbar ist es aber nicht. Die angegebenen Fixpacks machen auf jeden Fall Ihr OS/2-System »Year 2000 ready«, wie IBM es nennt. Schwierigkeiten in diesem Bereich dürften v.a. auf all jene zukommen, die OS/2 im geschäftlichen Umfeld einsetzen, da Jahreszahlen vom System nur mit zwei Stellen ausgegeben werden. Für Buchungsabwicklungen, die mit dem PC durchgeführt werden, kann es also zu Komplikationen kommen. Aber auch für den Kleinanwender kann es schwierig werden, etwa beim Abfragen eines Dateidatums.

Fragt man ein Dateidatum etwa mit REXX ab, so erhält man nur eine zweistellige Jahreszahl. Daher gibt es zwei neue REXX-Funktionen, die das Dateidatum vierstellig ausgeben. Die eine ist der neue Parameter QUERY TIMESTAMP der Funktion *Stream*; die andere taucht als Parameter L der Funktion *SysFileTree* auf. Wie man die zwei neuen Funktionen anwendet, zeigt das Codebeispiel auf der nächsten Seite. Die Ausgabe des kleinen REXX-Beispielprogramms DATUM.CMD gibt Abb.2 wieder. Zum Test des Programms stellten wir das Systemdatum auf das Jahr 2000. Das Ganze funktioniert. Auch über die WPS erhält man vierstellige Jahreszahlen in den Datumsangaben von Dateien. Nur ein einfaches *dir* gibt auf der Kommandozeile immer noch zweistellige Datumsangaben aus, denn statt 2000 findet man nur 00. Die Abfrage des Systemdatums über das DOS-API funktioniert -allerdings tat sie das schon vor einem »Jahr-2000-Problem«- wie gewünscht.

Was bitte macht das in €?
Wie Sie sehen, wurde OS/2 um das Euro-Zeichen (€) erweitert. Das neue Symbol erreicht man mit der Tastenkombination *AltGr+E*. Das Eurozeichen

```
/* REXX-Programm zum Abfragen von Dateidaten */
CLS
SAY ""
SAY "Geben Sie den Namen der Datei an, deren Datum"
SAY "abgefragt werden soll:"
SAY ""
PULL Dateiname
SAY ""
SAY STREAM(Dateiname, "C", "QUERY TIMESTAMP")

SAY ""
SAY "Geben Sie den Namen des Verzeichnisses an, dessen"
SAY "Dateien mit vierstelligem Dateidatum aufgelistet werden sollen:"
PULL Verzeichnisname
SAY ""
Call RxFuncAdd "SysLoadFuncs", "RexxUtil", "SysLoadFuncs"
Call SysLoadFuncs
Call SysFileTree Verzeichnisname, "Dateien", "L"
 DO i = 1 TO Dateien.0
  SAY Dateien.i
 END

SAY ""
SAY "Zum Beenden des Programms drücken Sie eine
SAY "beliebige Taste."
SAY ""
PULL
EXIT
```

Abb. 1: Dateidaten mit den neuen REXX-Funktionen abfragen

wurde zu verschiedenen Codepages hinzugefügt. Eine davon ist natürlich die Codepage 850, die standardmäßig installiert ist, und die man auch nicht ändern sollte.

Das Zeichen wurde den ATM-Schriftarten *Courier, Helvetica und Times New Roman*, den Bitmap-Fonts *Helv, System Monospace, System Proportional, System VIO und TmsRmn* sowie der Schriftart *Times New Roman MT 30*, dem Java 1.1 TrueType-Font hinzugefügt. Aber Vorsicht: Nicht alle diese Schriftarten erlauben einen korrekten Ausdruck des Zeichens. Sehen Sie daher von einem Satz in den Bitmap-Fonts ab, wenn Sie beabsichtigen, das Eurozeichen auszudrucken. Das liegt daran, daß die z.Zt. eingesetzten Drukker (noch) keine Bitmapschriftarten mit dem um das Eurosymbol erweiterte Zeichen ausdrucken können. Dort, wo das €-Zeichen auf dem Ausdruck sein sollte, findet man nur ein kleines **i** ohne Punkt vor, selbst wenn es zuvor auf dem Bildschirm richtig dargestellt wurde. Verwenden Sie daher eine ATM-Schriftart, am besten die *Helvetica* oder die *Times New Roman*, da Vektorfonts von jedem Drucker richtig ausgegeben werden. Die *OS/2 Only!* wurde z.B. in der Schriftart *Arial* gesetzt. Die €-Zeichen mit dem Font *Helvetica*. Da beide Schriftarten miteinander verwandt sind, bemerkt man den Unterschied kaum oder gar nicht. Ansonsten gibt es zum Gebrauch des neuen Zeichens nichts mehr zu sagen.

Das €-Zeichen wird übrigens nicht in WinOS/2 unterstützt. Legt man auf das Symbol der neuen Währung Wert, sollte man also mit OS/2-Programmen arbeiten. Mit dem neuen Zeichen ausgestattet, wird man natürlich dazu verleitet, es auch in eMails zu verwenden. Sofern man weiß, daß der Empfänger am anderen Ende OS/2 mit dem entsprechenden Fix verwendet, funktioniert das auch wie gewünscht. Allerdings sollte man hier das Zeichen dennoch lieber ausschreiben. Nicht jeder Computerbesitzer -mal

ganz allgemein ausgedrückt- ist eben auf dem neuesten Stand der Dinge (oder war es nie).

IBM gibt fernerhin an, daß die Unterstützung des neuen Zeichens bei der Installation neuer Graphikkarten verloren gehen kann. Meistens wird nämlich beim Einrichten einer neuen Graphikkarte die Datei DSPRES.DLL im Verzeichnis \OS2\DLL überschrieben. Damit kann das Euro-Zeichen verschwinden. Sollte dies der Fall sein, ersetzen Sie die DLL einfach durch die gefixte Version. Diese liegt im Verzeichnis \OS2\INSTALL\VGA. Dabei wird die Funktion des neuen Treibers übrigens nicht beeinträchtigt. Das Ersetzen können Sie also bedenkenlos vornehmen.

! Wenn Sie Ihr altes Java mit Java 1.1.4 aktualisieren möchten, so installieren Sie *zuerst* Java 1.1.4 und erst *dann* das Fixpack. Ansonsten kann die Unterstützung des Eurozeichens verloren gehen, und eine Neuinstallation des Fixpacks wäre erforderlich.

Removable Media Support

Über die Unterstützung von Laufwerken mit wechselbaren Medien haben wir schon in Band 1 der *OS/2 Only!* einiges geschrieben. Eingeführt wurde diese Unterstützung bereits einige FixPaks zuvor. Mit den hier vorgestellten Fixes kann man davon ausgehen, daß alles auch wie gewünscht funktioniert. Im Verzeichnis \OS2 findet man das kleine Dienstprogramm EJECT.EXE, mit dem man über die Kommandozeile Medien aller Wechsellaufwerke ausgeben kann, also auch die von CD-ROM-Laufwerken. Die Syntax ist einfach:

```
EJECT <Laufwerksbuchstabe>
```

Eine entsprechende Unterstützung dieser Funktion bietet auch die WPS: Hier verfügen Laufwerksobjekte, deren Medien wechselbar sind, über den neuen Menüpunkt *Datenträger Auswerfen*, der den gleichen Effekt erzielt. Weitaus angenehmer an den neuen Fixpacks ist aber, daß das Fehlerverhalten des Systems nicht mehr ganz so streng ist. Besonders bei der Arbeit mit der WPS war man vor ständigen Fehlermeldungen, daß das Laufwerk nach der

```
Datum.cmd

Geben Sie den Namen der Datei an, deren Datum
abgefragt werden soll:

d:\readme.1st

2000-04-01  17:18:52

Geben Sie den Namen des Verzeichnisses an, dessen
Dateien mit vierstelligem Dateidatum aufgelistet werden sollen:

f:\*

2000-04-03 18:05:58          24  -----   F:\chkdsk.log
2000-04-01 17:18:52      161029  A----   F:\README.1ST
1998-12-21 11:36:56           0  -D---   F:\Benchmark
1999-03-08 18:18:22           0  -D---   F:\bootos2
1998-12-20 17:30:54           0  -D---   F:\PM
1998-12-20 17:31:00           0  -D---   F:\System
1999-01-11 15:35:08           0  -D---   F:\tmp
1998-12-20 17:30:36           0  -D---   F:\WPS

Zum Beenden des Programms drücken Sie eine beliebige Taste
```

Abb. 2: Die Ausgabe des REXX-Programms zur Dateidatenabfrage

Ausgabe des Mediums nicht bereit sei, nicht so ganz sicher. Das hat sich glücklicherweise geändert. Mit der neuen Unterstützung der Wechsellaufwerke und den Veränderungen am Dateisystemcache kann man auch HPFS auf den Medien benutzen (was früher nicht ohne weiteres möglich war). Im Gegensatz zum FP 36 für Warp 3 bzw. FP 6 für Warp 4 verläuft das Wechseln HPFS-formatierter Medien jetzt ohne jede Schwierigkeit (auf einigen System konnten wir damit von Zeit zu Zeit Fehler beobachten, die nur mit einem anschließenden *chkdsk* behoben werden konnten). Voraussetzung für den Einsatz mit HPFS vorbereiteter Medien ist allerdings nicht nur, daß das Medium partitioniert ist, sondern auch, daß das Laufwerk ein SCSI-, IDE-, AT- oder ATAPI-Laufwerk ist bzw. für den Parallelport vom Hersteller durch einen ADD-Treiber unterstützt wird. Laufwerke, die OS/2 nicht selbst in die Laufwerksliste eintragen kann, sondern die erst nach Einbindung eines SYS-Treibers in die Liste hinzugefügt werden (das sind einige Laufwerke für den Parallelport), erlauben die Verwendung von HPFS-Medien nur mit den bereits schon vorgestellten Hilfsprogrammen und fallen nicht unter den neuen RMS des Betriebssystems.

In diesem Zusammenhang wollen wir noch einmal die allgemeine Untertützung für AT/ATAPI-Laufwerke erwähnen. Der Filtertreiber IBMATAPI.FLT bedurfte in vorangegangenen Fixpacks noch eines Fixes (s. Bd. 1). Mit den hier vorgestellten FPs entfallen derartige Schritte. Den Filtertreiber binden Sie wie die anderen FLT-Treiber via BASEDEV in Ihre CONFIG.SYS ein, sollten Sie die ATAPI-Unterstützung benötigen (etwa für IDE-Wechselplatten oder CD-Recorder). Wir werden in dieser Ausgabe im Zusammenhang mit ATAPI-CD-Recordern in unserem CD-Recorder-Test noch einmal auf den Filtertreiber zurückkommen.

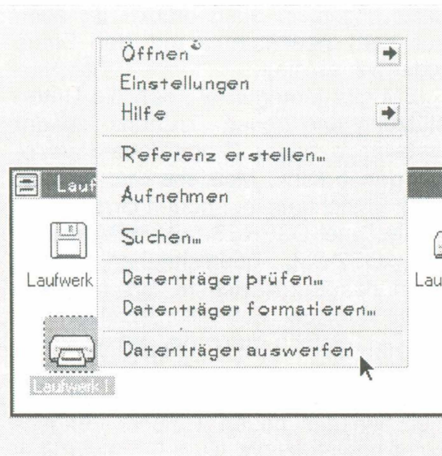


Abb. 3: Die Auswurffunktion der WPS

EIDE ganz groß

Wer schon einmal probiert hat, auf einem noch »jungfräulichen« System OS/2 auf einer EIDE-Festplatte zu installieren, die eine Kapazität über 2,1 GByte besitzt, wird festgestellt haben, daß **FDISK** unmögliche Kapazitäten bei der Partitionierung anzeigt. Diese können von 2 MByte bis zu mehreren tausend Terabyte reichen. Das Problem liegt darin, daß IBMS1506.ADD, der Treiber für die IDE-Festplattenunterstützung, nur mit Festplatten korrekt zusammenarbeitet, die eine Kapazität <= 2,1 GByte haben. Der IBMS1506.ADD in den neuen Fixpacks unterstützt hingegen Festplatten mit Kapazitäten über 8,4 GByte (mit dem FP35 für Warp 3 bzw. dem FP 6 für Warp 4 war die Grenze der unterstützten Kapazitäten noch auf 8,4 GByte festgelegt).

Sie sollten Ihre Installationsdisketten also auf jeden Fall mit den neuen Trei-

bern aktualisieren, damit Sie auch in den Genuß der neuen Unterstützung kommen. Verfahren Sie dazu folgendermaßen:

❶Kopieren Sie die Dateien IBM1S506.ADD, IBMIDECD.FLT und OS2DASD.DMD aus dem Fixpack auf die Diskette 1 der Installationsdisketten.

❷Öffnen Sie die Datei CONFIG.SYS, die Sie auf der Diskette 1 finden.

❸Überprüfen Sie, ob die SET-Anweisung SET COPYFROMFLOPPY=1 in der Datei zu finden ist. Wenn nicht, fügen Sie diese Anweisung hinzu. Sie bewirkt, daß die Treiber von den Installationsdisketten zu kopieren sind.

Führen Sie mit diesen Disketten die Installation durch. Wenn Sie dann *Benutzerdefinierte Installation* auswählen und FDISK aufrufen, werden die richtigen Kapazitäten angezeigt, und Sie können wie gewohnt fortfahren.

Tip: *Nachdem Sie über den Textteil der Installation hinaus sind, überprüfen Sie vorsichtshalber vor dem Neustart, ob die neuen Treiberdateien im Verzeichnis \OS2\BOOT liegen. Werfen Sie auch noch einmal rasch einen Blick in die CONFIG.SYS auf Ihrer Festplatte (mit TEDIT.EXE), um zu überprüfen, ob die IDE-Treiber auch richtig in das System eingebunden wurden.*

Stellen Sie jedoch zuvor einige Überlegungen zur Plattengeometrie an. Nicht jede Partition kann nämlich auch als Installationspartition bzw. als startbare Partition mit **FDISK** markiert werden. Das ist allerdings weder eine Einschränkung des Treibers oder des Dienstprogramms, sondern eine des **BIOS**, wonach eine startbare Partition innerhalb der ersten 1024 Zylinder der Platte liegen darf. Folgerichtig wird eine Partition, die jenseits des 1024sten Zylinders beginnt oder über den 1024sten Zylinder hinausreicht, nicht mehr als startbare **Partition** bzw. Installationspartition markiert werden können. Daher ist es ratsam zu bestimmen, wie groß die Partition überhaupt sein darf, wenn Sie als starbare Partition angegeben werden soll. Verfahren Sie dazu wie folgt:

❶Bestimmen Sie die Anzahl der Köpfe (Heads) und Sektoren (Sectors) Ihrer Festplatte. Meistens finden Sie diese Informationen direkt auf der Festplatte oder in einer Begleitinformation zum Gerät. Alternativ dazu können Sie auch das BIOS befragen oder OS/2 zum Abfragen der Information benutzen. Dazu versehen Sie den Eintrag:

BASEDEV=IBMS1506.ADD

mit dem Parameter /W und starten das System neu. Der Treiber gibt Ihnen dann die Plattengeometrie mit den oben gewünschten Informationen aus. Der Bootvorgang wird nach dem Anzeigen der Information unterbrochen, so daß Sie den Bildschirminhalt in Ruhe lesen können. Zu jeder Platte wird die Geometrie ausgegeben. Ganz links finden Sie die Anzahl von Cylinders, Heads und Sectors (in dieser Reihenfolge).

❷Berechnen Sie die den 1024 Zylindern entsprechende Größe mit:

$K_{1024} = (H_{HDD} \cdot S_{HDD})/2$,

worin K_{1024} die gesuchte Größe von 1024 Zylindern in MByte, H_{HDD} die Anzahl der Heads und S_{HDD} die der Sectors der Festplatte ist. Eine startbare Partition kann dann innerhalb dieses Speicherbereiches liegen, ihn aber nicht überschreiten. Runden Sie das Ergebnis immer nach unten ab, nicht nach oben.

❸Planen Sie von vornherein diesen zur Verfügung stehenden Speicherplatz für startbare Partitionen ein, insbesondere wenn Sie mehrere Betriebssysteme auf dem Rechner einsetzen möchten.

❹Richten Sie die Partitionen gemäß Ihrer Berechnungen ein, und verkleinern Sie sie ggfs. mit FDISK, sollten sie sich nicht als startbare Partitionen markieren lassen.

Grundsätzlich gilt:

1. Die beschriebene 1024-Zylinder-Einschränkung gilt *sowohl* für IDE(EIDE)- *als auch* für SCSI-Platten. Beachten Sie daher *immer* die Geometrie der Festplatte.
2. Innerhalb des nach dem oben angegebenen Verfahren bestimmten Speicherbereiches können beliebig viele Partitionen als startbare Partitionen eingetragen werden.
3. FAT-Partitionen können unter OS/2 eine Größe von max. 2,1 GByte haben; HPFS-Partitionen hingegen können max. 64 GByte groß sein.

Tip: *Wenn Sie genau wissen, daß Sie mehrere Betriebssysteme auf einer Festplatte installieren möchten, so greifen Sie von vornherein zu einer Platte, deren Geometrie einen möglichst großen Speicherbereich für das Einrichten startbarer Partitionen gestattet.*

Nach erfolgreicher Installation des Betriebssystems installieren Sie am besten zu allererst das entsprechende Fixpack, bevor Sie mit der Konfiguration der WPS und dem Einrichten der Anwendungen fortfahren.

GRADD?

Mit Warp 4 lieferte IBM eine neue Architektur für Graphikkartentreiber namens *Graphics Adapter Device Drivers (GRADD)*. Die neue Architektur erlaubt nicht nur die leichtere Entwicklung neu-

Zur Installation eines der GRADD-Treibers geben Sie ein:

setup <Treibername>

wobei <Treibername> den Typ es zu installierenden Treibers angibt und folgende Werte annehmen kann:

ATI
ATI Rage/Mach 64

CHP
Chips & Technology 6555X

MGA
Matrox Millenium/Mystique

VIRGE
S3 Virge

S3
S3 864/Trio

VGA
VGAGRADD Standard VGA

GEN
GENGRADD Standard SVGA

Abb. 4: Parameter zum GRADD-Setup

```
***** There is no vesa bios support on your graphics card.
***** You cannot run gengradd on this machine.
***** Restart the machine, hit alt-F1 to set it back to VGA.
***** If this screen appears again, reboot, hit alt-F1 to command prompt,
***** make sure genpmi.dll is removed from \os2\dll
```

Abb. 5: Fehlermeldung des GENGRADD-Treibers

er OS/2-Treiber für Graphikkarten; sie bietet auch Möglichkeiten zu Leistungssteigerungen unter dem PM, sofern die neuen Treiber auf dieser Architektur aufsetzen. Diese Leistungssteigerung betrifft die beschleunigte und unbeschleunigte Graphikdarstellung sowie die Wahl größerer Farbtiefen und Auflösungen. Dabei ist die GRADD-Architektur optimal an die Speicherverwaltung des Systems angepaßt und erlaubt die einfache Benutzung des Speichermanagements seitens neuer Treiber unter dem Betrieb der 32-bittigen Graphikarchitektur von OS/2. Ein GRADD-Paket kann man auf den IBM-Webseiten des TA Mainz in der momentan aktuellen Version 0.77 beziehen. In diesem Paket sind verschiedene GRADD-Treiber enthalten, die Sie *Abbildung 4* entnehmen können.

Neben einigen gerätespezifischen Treibern findet man in dem GRADD-Treiberpaket aber auch VGA und SVGA GRADD-Treiber. Fixes und weitere Treiber bekommt man auch auf dem IBM-Server *http://service5.boulder.ibm.com.*

Sie können diese Treiber auf Ihrem System einsetzen, allerdings setzt dies voraus, daß Ihre Graphikkarte ein nach den VESA-Spezifikationen konformes BIOS besitzt. Lediglich der VGAGRADD-Treiber arbeitet auch mit Karten ohne ein solches BIOS zusammen. Die im GRADD-Paket von IBM enthaltenen Treiber können nur mit einem Warp 4 mit FP 5 oder höher und auch mit Warp 3, hier allerdings mit FP 38 oder höher betrieben werden (ob ein Betrieb der Treiber auch mit niedrigeren FP-Levels möglich ist, haben wir nicht getestet).

Die Treiber bedürfen eines speziellen Installationsprogrammes, das im GRADD-Paket enthalten ist. Zur Installation dient das Programm *setup*, dem man als Parameter den zu installierenden Treiber nennt. Die Syntax entnehmen Sie ebenfalls *Abbildung 4.* Zur Installation der Treiber:

❶ Erstellen Sie ein temporäres Verzeichnis und dekomprimieren Sie dort das Treiberpaket.

❷ Öffnen Sie ein OS/2-Fenster und wechseln Sie in das Verzeichnis, in dem die entpackten GRADD-Treiber liegen.

❸ Rufen Sie *setup* mit dem entsprechenden Parameter auf (siehe Abb. 4). Um z.B. den generischen SVGA-Treiber zu installieren, geben Sie ein:

setup gen

Das Installationsprogramm weist Sie nach Abschluß der Installation darauf hin, daß Sie das System neu starten müssen.

❹ Geben Sie den Befehl *shutdown* ein, und starten Sie das System nach Beendigung des Systemabschlusses neu.

Der GENGRADD-Treiber funktionierte auf allen Testsystemen, vorausgesetzt, die Graphikkarten verfügten über ein VESA-konformes **BIOS**. Ist das nicht der Fall, so erhält man beim Start des

Treibers die in *Abb. 5* angegebene Fehlermeldung. Dann hilft nur ein Zurücksetzen auf Standard-VGA und die Reinstallation des zuvor eingesetzten Graphiktreibers. Der generische VGA-Treiber arbeitete ausnahmslos auf allen Systemen. Allerdings bringt er so gut wie gar nichts, weil es fraglich ist, wer seinen Rechner im **VGA**-Modus laufen lassen möchte (also mit 16 Farben und einer Auflösung von 640x480 Bildpunkten). Hier funktionieren auch die normalen Standard-VGA-Treiber des Betriebssystems, die, wie sich zeigte, auch nicht langsamer sind als der GRADD-Treiber.

Was den GENRADD-Treiber angeht, so ist auch dieser nur eine Notlösung. Er bietet zwar höhere Auflösungen und Farbtiefen, so daß man zumindest schon einmal in den Genuß des SVGA-Standards kommt; aber der Treiber unterstützt nicht die Beschleunigungsfunktionen des Graphikprozessors, so daß man den besonderen Vorteil einer **Accelerator-Karte** nicht ausschöpfen kann. Hieraus folgt, daß GENRADD auf schnellen und gut konfigurierten Systemen eine bessere Leistung zeigt als auf solchen, bei denen die CPU höher belastet ist (da ohne die Arbeit des Graphikprozessors auf der Karte, die CPU alle Zeichenfunktionen übernimmt). Dennoch ist es besser als nichts. Denn da für viele Karten keine OS/2-Treiber existieren, bleibt v.a. Umsteigern auf OS/2 zunächst nur einer der Standardtreiber, und ob der (mal abgesehen vom VGA-Treiber) anstandslos seinen Dienst tut, bleibt fraglich. Mit GENRADD hat man wenigstens ein Provisorium, und das ist gar nicht mal so schlecht.

Wenngleich auch die Leistung aus dem oben genannten Grund ab und an nicht so sehr beeindruckt (v.a. bei Scrolling- oder typischen Zeichenoperationen), so werden Bilder doch farbrichtig dargestellt, v.a. wenn man sie skaliert. Auch die Wiedergabe von Fonts und kleineren Details erfolgt sauber und z.T. bedeutend besser als mit den eigentlich für die Graphikkarte vorgesehenen Treibern. Auf der anderen Seite wird das Bild wieder nicht ganz so rosig, wenn man bedenkt, daß der Bildaufbau z.T. überhaupt nicht mehr korrekt ist, und auch die Farbwiedergabe der Systemcontrols von Zeit zu Zeit wechselt. Aber die Fähigkeit, eine im großen und ganzen doch gut gelungene Standard-SVGA-Unterstützung für nahezu alle auf dem Markt gängigen Graphikkarten anzubieten, muß man dem GENRADD-Treiber auf jeden Fall lassen.

Ganz anders hingegen sieht es mit den adapterspezifischen Treibern aus. Wir testeten nur den MGAGRADD mit einer Matrox Millenium, einer recht populären Karte, für die es auch OS/2-Treiber gibt. Zunächst einmal verrichtet der Treiber wie beabsichtigt seinen Dienst. Weiterhin zeigt er gute Leistungen, die mit dem herstellereigenen Treiber etwa auf einer Ebene liegen, bietet aber eine größere Auswahl an Auflösungen und Farbtiefen. Probleme während des Testes, wie wir sie beim GENRADD antrafen und zuvor beschrieben, traten nicht auf. Hier bietet sich mit den GRADD-Treibern eine interessante Alternative, so daß zu hoffen bleibt, IBM wird noch weitere Treiber entwickeln. Zumindest bewährt sich die zugrundeliegende Architektur. Ausprobieren kann man die neuen Treiber also, wenn man eine Graphikkarte hat, für die eine spezielle Unterstützung vorliegt. GENRADD sollte man als Notlösung im Auge behalten. VGARADD kann man vernachlässigen.

Das 32-bit chkdsk

Auch wenn es schon in früheren Fixpacks auftauchte, wollen wir es kurz

erwähnen, da sonst nirgends eine Dokumentation (außer in den Readme-Dateien) zu finden ist: Das neue *Check-Disk*, dem man als *chkdsk* auf der Kommandozeile und auf der WPS (dahinter verbirgt sich das Programm *pmchkdsk.exe,* das nur über die **WPS** gestartet werden kann) unter den Menüpunkten *Datenträger prüfen* der Laufwerksobjekte begegnet.

Gegenüber dem Vorgänger bietet das neue *chkdsk* einige Verbesserungen, die jedoch nur auf Laufwerken, die mit HPFS formatiert wurden, zum Vorschein treten. Dies ist neben einer verbesserten Fehlererkennung auch eine höhere Ausführungsgeschwindigkeit, die allerdings nicht so stichhaltig zu Tage tritt. Besonders interessant ist jedoch die Log-Funktion. *Chkdsk* erstellt während seiner Tätigkeit eine Protokolldatei, die man im Stammverzeichnis eines Laufwerkes unter dem Namen *chkdsk.log* findet. Deren Inhalt kann man sich mit dem Programm *pmchklog.exe* betrachten. Sie erhalten dieses Programm auf dem IBM-Fixpack-Server oder unserer Homepage. IBM bietet für das Werkzeug keinerlei Support (und wehrt sich auch ausdrücklich gegen Anfragen), so daß wir Ihnen im *Know How* Kapitel die Funktionsweise des Programms zeigen und erklären, wie der Output von *chkdsk* zu deuten ist. Sie finden dort auch eine Beschreibung der weiteren neuen Features des kleinen aber wichtigen Dienstprogrammes und wie man es für unterschiedliche Zwecke einsetzt. Hier bleibt nur noch zu sagen, daß Sie *chkdsk* wie immer verwenden, also über die WPS oder durch Eingabe des Befehls *chkdsk* von der Kommandozeile aus. Alles weitere erfahren Sie später.

Zum Schluß

Es bleibt ein guter wie erfreulicher Eindruck. Da ohne die Fixpacks, die wir hier vorstellten, gewisse Programme nicht laufen und bestimmte Hardwarekomponenten im Dunkel des Nichterkennens verschwinden, muß man ohnehin zu diesen Fixpacks greifen. Es lohnt sich. ❑

Driving all over

Treiber und OS/2 - zwei Welten treffen aufeinander? Wie man's nimmt. Für einige Bereiche sieht man sich gut mit Treibern versorgt, z.B. für SCSI-Adapter. Hier ist es sogar der Marktführer Adaptec, der seine Adapter traditionell mit einer guten OS/2-Treiberunterstützung ausstattet. Aber wie sieht es aus mit Treibern für Graphikkarten? Zur Zeit gibt es nur noch wenige Hersteller, die Ihre Karten mit eigenen OS/2-Treibern ausstatten, etwa Elsa und Matrox (wobei die Treiberunterstützung von Elsa weitaus besser ist). So war der OS/2-Benutzer in seinen Auswahlmöglichkeiten den Graphikadapter betreffend bislang stark beeinträchtigt. Das könnte sich aber ändern, seit *SciTech* den *DisplayDoctor*, eine von DOS und Windows her bekannte Software, auf OS/2 portiert. Um es vorwegzunehmen: Mit dieser Treibersoftware ausgestattet, kann man unter OS/2 auch all jene Graphikkarten betreiben, für die normalerweise keine OS/2-Treiber verfügbar sind. Das hört sich unglaublich an, aber entspricht der Wahrheit. Mit dem *DisplayDoctor* soll das Betreiben beliebiger Graphikkarten unter OS/2 möglich sein. Was das für den OS/2-Benutzer bedeutet, ist klar: Unabhängigkeit in der Wahl seiner Graphikkarten. Sollte der *DisplayDoctor* das halten, was *SciTech* verspricht, dürfte das sicher eine Art Revolution der Treiberverfügbarkeit für OS/2 sein. In Bd. 1 der *OS/2 Only!* haben wir ja bereits eine

Ankündigung gemacht. Nun haben wir uns das interessante Produkt in einer Betaversion (Beta 3) einmal näher angesehen.

Was ist der DisplayDoctor?

Der *SciTech DisplayDoctor* für OS/2 -im folgenden kurz *SDD/2* genannt- basiert auf dem *SciTech Nucleus Graphiktreiber*, einem plattformunabhängigen Treiber, der unter OS/2 seine Runtime Umgebung mit der *GRADD Graphiktreiberarchitektur* von IBM findet. Da der Nucleus Treiber unter OS/2 auf dieser Architektur aufbaut, ist eine hohe Leistung des SDD-Treibers sichergestellt. Der *SciTech Nucleus* wird durch seine Plattformunabhängigkeit ausgezeichnet: Durch den Treiberkern wird es möglich, die gleichen *SciTech Treiber* unter OS/2 einzusetzen, die auch unter Windows und DOS laufen. *SciTech* hat angekündigt, den gesamten Treibercode zur Unterstützung der auf dem Markt gängigen Graphikchips in den neuen *Nucleus Treiber* zu übertragen. Damit wird die Unterstützung fast aller Graphikadapter unter OS/2 gewährleistet, das sind weit über 300 verschiedene Modelle. Zur Zeit ist erst die Beta 3 des Produktes verfügbar. Eine Liste der momentan unterstützten Graphikkarten finden Sie in *Abbildung 6*. Auf jeden Fall unterstützt der *SDD/2* keine Graphikkarten mit einem Graphikspeicher von **weniger** als 512 KByte - aber diese Einschränkung dürfte wohl irrelevant sein.

Da der *Nucleus Treiber* eine neue Entwicklung von *SciTech* ist, kann man davon ausgehen, daß *SciTech* auch in Zukunft auf den plattformunabhängigen Weg setzen wird, was OS/2 natürlich zugute kommt, denn neue Treiber sind dann für alle Plattformen verfügbar.

Wie bereits angeführt, bietet die GRADD-Architektur erhöhte Leistungen für Treiber, die auf ihr aufsetzen (wie die Implementierung der *SciTech-Treiber*). Ein GRADD-Paket kann man neben den bereits genannten Quellen auch vom

Alliance
ProMotion 6422, ProMotion AT24, ProMotion AT25, AT3D

ATI
Mach64 CT, Mach64 ET, Mach64 VT, 3D Rage, Mach64 VTB, 3D Rage II, 3D Rage II+, 3D Rage IIC, 3D Rage Pro, 3D Rage LT Pro

Cirrus Logic
5434, 5436, 5440, 5446, Laguna 5462, Laguna 5464, Laguna 5465

Cyrix
Media GX, Media GXi, Media GXm

InteGraphics
IGA1680, IGA1682, IGA1683, CyberPro 2000, CyberPro 2010

Intel
i740

Macronix
86250, 86251

Matrox
MGA Millennium, MGA Millennium II, MGA Mystique, MGA Mystique 220, MGA G100, MGA G200

Philips
9710, 9712

S3
Trio32, Trio64, Trio64V+, Trio64UV+, Trio64V2/DX, Aurora64V+, Virge, Virge/DX/GX, S3 Virge/VX, Virge/GX2, Trio3D

SiS
6202, 6205, 6215, 5595, 5597/5598, 6326

Tseng Labs
ET6000, ET6100, ET6300

Abb. 6: Von *SDD/2 Beta 3* unterstützte Graphikkarten

SciTech-FTP-Server beziehen, wenn man sich die Beta des *SDD/2* besorgt.

SDD/2 stellt auch einen GRADD-Treiber zur Verfügung, der mit dem *SciTech* Nucleus Treiber zusammenarbeitet: SDDGRADD. Dieser Treiber kann demnach nur dann Ihre Hardware unterstützen, wenn das **BIOS** der Graphikkarte auch die GRADD-Architektur unterstützt. Daher kann es durchaus vorkommen, daß der *SDD/2* zwar mit einem bestimmten Chipsatz zusammenarbeiten kann; der Treiber aber dennoch nicht läuft, wenn das BIOS des Graphikadapters nicht mitspielt: Ohne funktionierende GRADD-Architektur kein *SDD/2*. Eine Garantie, daß Sie auch ältere Karten mit dem *SDD/2* betreiben können, kann es daher nicht geben, so daß hier nur Ausprobieren hilft. Bei neueren Adapterkarten dürften aber keine Schwierigkeiten auftreten, da sich die **VESA**-Standards längst auf dem Markt durchgesetzt haben.

Installation

Selbst wenn Ihre Graphikkarte nicht in der oben wiedergegebenen Kompatibilitätsliste aufgeführt ist, sollten Sie den *SDD/2* einfach einmal testen, vielleicht haben Sie bereits mit der Betaversion Glück. Besorgen Sie sich zunächst die *SDD/2-Beta*. Auf unserer Homepage unter der Rubrik Treiber finden Sie einen Link. Wenn Sie sich die Dateien direkt vom *SciTech-FTP-Server* holen möchten, benutzen Sie diese Adresse:

ftp.scitechsoft.com/sdd/beta/os2/

Dort finden Sie die beiden Archive:

sddos2_70_b3.zip und
ibmgradd.zip

Das Archiv *ibmgradd.zip* enthält die GRADD-Architektur in der Version 0.77 mit den aktuellen IBM-GRADD-Treibern, die wir Ihnen bereits kurz vorstellten. Für alle später evtl. noch folgenden Betaversionen benutzen Sie ebenfalls diese Datei, also genügt ein Download. Das Archiv *sddos2_70_b3.zip* enthält die eigentliche Beta 3 des *SDD/2*. Nach dem Dateitransfer installieren Sie den *SDD/2* wie folgt:

❶ Erzeugen Sie ein Verzeichnis auf Ihrem Bootlaufwerk, z.B. C:\SDD, und entpacken Sie die beiden Archive direkt in dieses Verzeichnis. Die GRADD-Dateien finden Sie dann in einem Unterverzeichnis namens IBMGRADD, etwa:

<Boot.-LW>:\SDD\IBMGRADD

SciTech macht darauf aufmerksam, bei Verwendung von PKZIP den Parameter -d beim Entpacken anzugeben.

❷ Öffnen Sie ein OS/2-Fenster, wechseln Sie in das SDD-Verzeichnis und starten Sie die Batchdatei setup.cmd. Sollte Ihr Bootlaufwerk nicht c:\ sein, öffnen Sie setup.cmd zunächst mit einem Editor und ersetzen Sie die Laufwerksbuchstaben Ihrem Bootlaufwerk entsprechend, da dort nur absolute Laufwerksangaben mit c:\ als Laufwerk stehen. Ersetzen Sie daher die einzelnen Anweisungen, z.B.:

mkdir c:\os2\drivers\nucleus\config durch
mkdir <LW>:\os2\drivers\nucleus\config,

wobei <LW> Ihr Bootlaufwerk ist.

Verfahren Sie ggfs. auch mit der Datei sddgradd.cmd so, in der Sie den Befehl:

copy c:\os2\svgadata.pmi c:\os2\svgadata.bak

entsprechend abändern, indem Sie wieder Ihr Bootlaufwerk statt c:\ angeben.

❸Nachdem Sie *setup.cmd* evtl. entsprechend geändert und ausgeführt haben, öffnen Sie Ihre CONFIG.SYS und fügen Sie den SDD-Pfad in die PATH-Anweisung ein, z.B.:

<LW>:\SDD

wobei <LW> wieder der Buchstabe Ihres Bootlaufwerkes ist. Fügen Sie außerdem folgende DEVICE-Anweisung hinzu:

DEVICE=<LW>:\OS2\SDDHELP.SYS

Auch hier ist <LW> wieder Ihr Bootlaufwerk. SDDHELP.SYS ist ein Treiber, den die *SciTech-Testprogramme*, mit denen Sie noch arbeiten werden, benötigen. Mit setup.cmd wurde diese Datei in das \OS2-Verzeichnis kopiert. Die PATH-Anweisung ist nicht nötig, aber empfohlen, damit Sie die später auszuführenden Testprogramme starten können, ohne in das SDD-Verzeichnis zu wechseln. Wir gehen in den folgenden Schritten davon aus, daß Sie die Eintragung in den PATH vorgenommen haben.

❹Führen Sie einen Systemabschluß aus und starten Sie den Rechner anschließend neu.

❺Öffnen Sie nach dem Neustart einen OS/2-Gesamtbildschirm und geben Sie folgenden Befehl ein:

gatest

Damit starten Sie das Programm GATEST, das neben anderen Werkzeugen im SDD-Verzeichnis liegt, und mit dem Sie überprüfen, ob der SDD/2 mit Ihrer Graphikkarte zusammenarbeitet.

❻Nach dem Start von GATEST finden Sie ein Menü vor, durch das Sie durch die Auswahl von Ziffern navigieren. Wählen Sie hier den Interactive Test, und wählen Sie in dem daraufhin angezeigten Textmenü die Farbtiefe und Auflösung aus, mit der Ihre Graphikkarte arbeiten soll. GATEST zeigt dann, je nach gewählter Farbtiefe, unterschiedliche Testbilder. Werden sie korrekt angezeigt, dürfte Ihre Graphikkarte mit dem *SDD/2* zusammenarbeiten. Verlaufen die Tests negativ, können Sie von einer weiteren Installation des SDD/2 i.d.R. absehen.

❼Beenden Sie GATEST durch Auswahl der entsprechenden Menüeinträge und geben Sie auf der Kommandozeile den Befehl

setvga

ein, um Ihr System auf Standard-VGA zurückzusetzen. Führen Sie einen Systemabschluß durch und starten Sie den Rechner anschließend neu.

❽Nach dem Neustart öffnen Sie ein OS/2-Fenster und geben dort den Befehl:

sddgradd

ein. Dadurch wird die Batch-Datei *sddgradd* im SDD-Verzeichnis gestartet. Sie ruft das IBM-GRADD Setup-Programm mit dem Parameter gen auf, das daraufhin den IBM GEN-GRADD-Treiber installiert. Anschließend kopiert das Batch die SDD/2-Dateien über die der generische

SVGA-Unterstützung. Die Installation der *SDD-GRADD-Treibers* ist damit abgeschlossen. Führen Sie aber **noch keinen** Systemabschluß durch.

⑨ Öffnen Sie Ihre CONFIG.SYS und tragen Sie den SDDGRADD-Treiber als aktuellen Graphiktreiber ein. Ändern Sie dazu die vom GRADD-Setup gesetzte Umgebungsvariable:

SET C1=GENGRADD,SBFILTER

auf folgenden Wert:

SET C1=SDDGRADD

Die Variable C1 sollte dabei wie folgt gesetzt sein:

SET GRADD_CHAINS=C1

Fügen Sie außerdem die Anweisung:

SET GREEXT=SDDGREXT

hinzu, wenn für die Textdarstellung auf der Arbeitsoberfläche eine höhere Fontgröße benutzt werden soll. In der Betaversion läßt sich diese Funktion zunächst nur über diese Umgebungsvariable setzen.

⑩ Sichern Sie die CONFIG.SYS, führen Sie **erst jetzt** einen Systemabschluß durch, und starten Sie das System neu. Danach läuft die Graphikkarte mit dem *SDD/2-Treiber* mit einer Farbtiefe von 256 Farben und einer Auflösung von 640 x 480 Bildpunkten.

Die weitere Konfiguration nehmen Sie OS/2-konform über das System-Notizbuch auf der Seite Bildschirm durch.

Läuft er oder läuft er nicht?

Da wir erst kurz vor Redaktionsschluß die Mitteilung von *SciTech* bekamen, daß die Beta des *SDD/2* verfügbar ist, blieb uns nicht viel Zeit für einen Test. Wir konnten daher den neuen Treiber nur mit den in unseren Rechnern eingesetzten Karten testen, für die natürlich bereits eine OS/2-Unterstützung vorliegt. In diesem Artikel stellen wir daher nur kurz den Einsatz des *SDD/2* auf einer Cirrus Logic Karte mit 5434 Chipsatz vor. Ein gründlicher Test ist also einem folgenden Band vorbehalten. Wir bereiten einen solchen *SDD/2*-Test mit unterschiedlichen Graphikkarten vor, vor allem mit solchen, für die keine her-

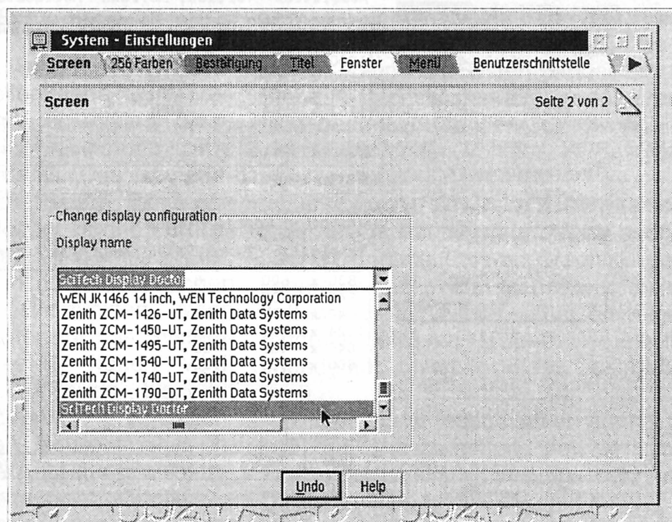


Abb. 6: Als aktuellen Monitor den SDD/2 auswählen

stellereigenen Treiber existieren und die OS/2 auch von Haus aus nicht unterstützt (wie es bei den 5434-Karten von *Cirrus Logic* der Fall ist, die wir zum Kurztest benutzten).

Der Betrieb des SVGRADD bereitete keine Probleme. Nach der beschriebenen Installation, die noch etwas mühsam ist, da die Beta noch kein eigenes Setupprogramm besitzt, findet man im *System-Einstellungsnotizbuch* auf der Seite *Bildschirm* (die nach der Installation der GRADD-Treiber und des *SDD/2 Screen* heißt), auf Seite 2 den *SDD/2* unter der Liste unterstützter Monitore aufgeführt. Wählen Sie hier den *SciTech DisplayDoctor* aus der dargestellten Liste. Mit einem Klick auf den Button View current configuration erhalten Sie eine Aufstellung der vom *SDD/2* gebotenen Auflösungen und Farbtiefen (siehe *Abb. 7*). Um Auflösung und Farbtiefe für Ihr System festlegen zu können, wechseln Sie auf Seite 1 der Screen-Seiten im Notizbuch und wählen die gewünschte Konfiguration aus. Bedenken Sie, daß der *SDD/2* zwar eine ganze Anzahl an Bildschirm-Konfigurationen bietet, diese aber nicht immer genutzt werden können, wenn die Grenzen der eingesetzten Hardware dies nicht zulassen. Das gilt natürlich nicht nur für die Graphikkarte, sondern auch für den Monitor: Unterstützt die Hardware die möglichen Auflösungen und Farbtiefen nicht, dann können Sie diese natürlich auch nicht wählen. Das ist grundsätzlich mit allen Treibern so, also informieren Sie sich zuvor, in welchem Rahmen Graphikkarte und Monitor zusammenarbeiten. Wie

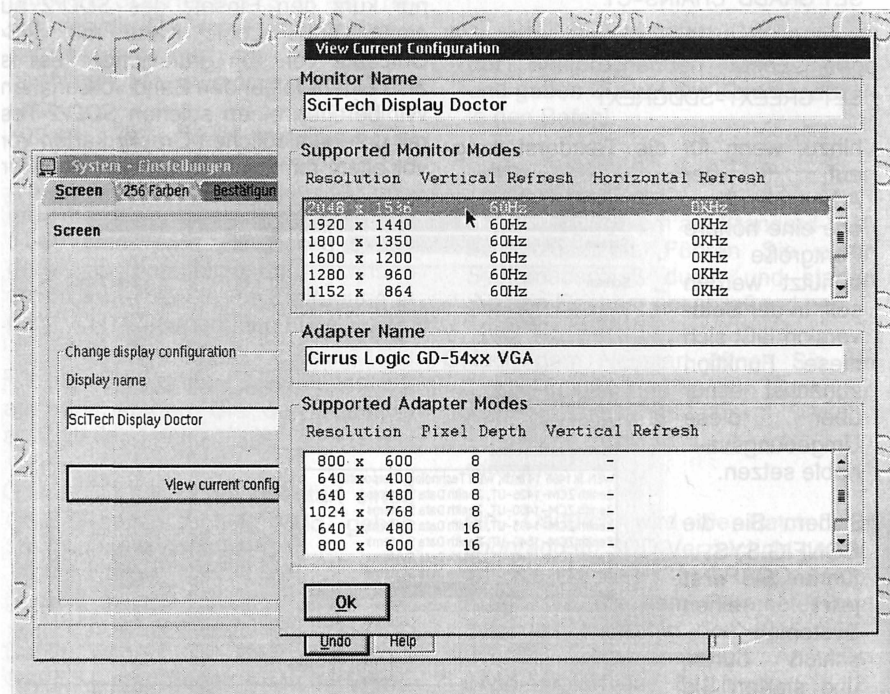


Abb. 7: Auflösungen und Farbtiefen des SDD/2

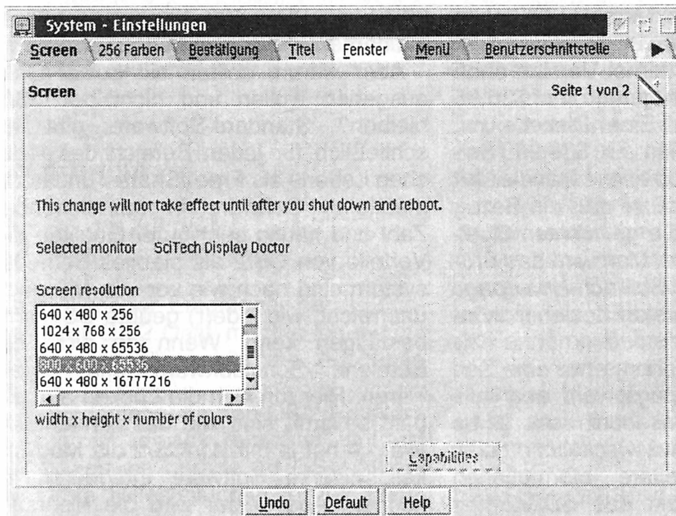


Abb. 8: Die gewünschte Auflösung und Farbtiefe festlegen

die Selektion der Bildschirmauflösung aussehen kann, zeigt *Abb. 8*.

Beim Betrieb mit einer mittelmäßigen und alten Karte wie einer *Cirrus Logic* mit 5434 Chipsatz wird der große Vorteil des *SDD/2* natürlich nicht so deutlich. Bei modernen Karten aber, die hohe Auflösungen und Farbtiefen ermöglichen, schon. Denn wenn für solche Graphikadapter herstellereigene OS/2-Treiber verfügbar sind, so bieten diese meist nicht die gleichen Leistungen wie ihre Windows-Pendants. Das ist natürlich ärgerlich. Der *SDD/2* hingegen unterstützt die Auflösungen und Farbtiefen bis zu den Grenzen der Hardware. Das sind momentan noch Auflösungen bis zu 1600x1200 Bildpunkten. Wenn *SciTech* eigene Seiten für das System-Notizbuch fertiggestellt haben wird, kann man die Auflösung bis auf 2024x1536 Bildpunkte einstellen. Berücksichtigt man schließlich noch, daß *SciTech* die Anzahl der unterstützten Karten noch vergrößern möchte, wie wir weiter oben ausführten, steht der OS/2-Anwender den Benutzern der Windowsderivate in puncto Graphikkarten in nichts nach.

Was die Leistung des SDDGRADD-Treibers angeht, so waren wir zufrieden. Farben wurden, v.a. bei einem Darstellungsmodus von 256 Farben, im Gegensatz zum im Lieferumfang enthaltenen Treiber für die eingesetzte Graphikkarte korrekt dargestellt. Auch die Geschwindigkeit beim Ausführen typischer Zeichenfunktionen entsprach der eines Adapters mit einem Graphikprozessor (wir beobachteten ja die Nachteile des generischen SVGA-GRADD-Treibers und berichteten darüber). Im übrigen traten auch keine Fehler im laufenden Betrieb auf, selbst dann nichts, wenn das System länger als 12 Stunden intensiv genutzt wurde. Wenn man bedenkt, daß der *SDD/2* noch in einer Betaversion vorliegt, ist das eine reife Leistung.

Zum Test des Treibers auf Systemen, deren Graphiksubsystem besser ausgebaut ist, und mit anderen Graphikkarten, konnte es leider nicht mehr kommen, so daß wir es bei diesem eher flüchtigen Eindruck belassen müssen. Wie bereits angedeutet, holen wir diese Tests aber nach.

Erfreulich war jedoch schon einmal zu erfahren (und zu sehen), daß der neue Treiber funktioniert - und das sehr gut. Die Betaversion lohnt also auf jeden Fall einen Testlauf, und wir erwarten mit

Spannung die erste Version. Übrigens ist der *SDD/2* auch erschwinglich. Bezieht man eine Internet-Version ohne Handbuch, wird man gerade mal $36,95 ausgeben müssen. Eine Diskette mit Handbuch erhält man für $39,95. SciTech wird auch europäische Händler mit dem *SDD/2* beliefern, so daß ein Bezug der Software über die gewohnten Quellen möglich sein wird. Man wird das Produkt direkt über die *SciTech-Homepage* als 21-Tage-Testversion beziehen können.

Weitere Informationen, etwa eine Seite dokumentierter Bugs, wird ebenfalls über das WWW einsehbar sein. Unter dem URL: *http://www.scitechsoft.com* wird man bald weitere Informationen erhalten. Der Auftakt des *SDD/2* war gut. Auf weiteres darf man also gespannt sein. □

Was ist dran...?

Martin Gratz

Gerüchte, die man auch auf der CeBIT nicht dementieren wollte, sprechen von einer Vielzahl neuer privater OS/2-Nutzer. Im Zuge von Millenium-Projekten und ähnlichem sollen bei Banken und Versicherungen in nächster Zeit Zehntausende von Computern ausgemustert werden. Diese gehen dann zu einem Spottpreis an das Personal. Jetzt der Clou: Mit welchem Betriebssystem sind die wohl ausgestattet?

Allerdings gab mancher Gesprächspartner den bedenkenswerten Hinweis, daß die Mehrzahl wohl auf schnellstem Wege den Rechner auf Windows umstellen wird. »Egal, ob mit oder ohne legal erworbener Lizenz: Software für OS/2 gibt es eben nicht bei jedem Händler - für Windows hat's der Nachbar.« Diese Computer-Neubesitzer müßten also eine Vollversion Windows 98 für rund DM 498,- kaufen, um legal mit dem Rechner zu arbeiten.

Aber warum sollten sie soviel Geld ausgeben wollen und nicht bei OS/2 bleiben? Standard-Software gibt es schließlich für jeden Bereich des täglichen Lebens als Free-/Share-/ und kommerzieller Software in ausreichender Zahl und häufig sehr guter Qualität; die Vorteile von OS/2 als stabiles Betriebssystem sind nach wie vor von MS noch unerreicht, wie jede(r) geübte OS/ler(in) bestätigen kann. Wenn da nicht die Bereiche Schulsoftware und Spiele wären. Hier tun sich die Lücken auf, die bald so groß sind wie das Rentenloch. *Warp 4* hat ja mit *WinOS/2* die Möglichkeit, diverse Software integrieren zu können. Aber leider wird die Mehrzahl der Schulsoftware bereits auf Windows umgestellt. Aktuelle kommerzielle Spiele erscheinen prinzipiell nur noch für die verbreiteten MS-Betriebssysteme.

Bei meinem Rundgang über die CeBIT, der allerdings nur stichprobenartig ablief, wollte außer den traditionell mit OS/2 verbundenen Herstellern auch keiner mit neuer Software oder Gerätetreibern auf den OS/2-Zug aufspringen. Stets das gleiche Argument: Es gäbe zuwenig (zahlende) Nutzer. Für LINUX, sogar BeOS hätte man allerdings schon etwas - oder wenigstens geplant. Aber OS/2? Das würde doch keiner privat benutzen! (Nur: Wieviele Privatanwender haben denn schon mal was von BeOS gehört?) Auf das obige Gerücht angesprochen, kam mehr oder weniger direkt die Anregung, doch schon mal eine Bestellung für OS/2-Versionen (in ausreichend hoher Zahl) abzugeben. Dann wäre das kein Problem.

Selbst auf dem IBM-Stand mochte man keine definitive Auskunft geben, ob als zukünftige Client-Version noch ein »Zwischenrelease Warp 4.5« oder ein um die Serverfunktionen abgespecktes

»Aurora« vielleicht noch in diesem Jahr erscheint. Aber an manchen Gerüchten soll ja bekanntlich was dran sein... ❑

Leserbriefe

Tips für die Zukunft

Rüdiger Strothmann via eMail

Als zufriedener Empfänger der 1. Ausgabe der *OS/2 Only!* möchte ich hiermit Anregungen für zukünftige Ausgaben geben.

Zumindestens für den OS/2-Privatanwender ist diese Zeitschrift äußerst wichtig, da sie gerade für OS/2-Anfänger erste Hilfestellungen leisten kann. Ein halbes Jahr nach der Einstellung der *Inside* kommt nun eine Fachzeitschrift auf den Markt, die in Deutschland ihresgleichen sucht. Über die Qualität irgend welcher Win-Zeitschriften möchte ich mich nicht auslassen. Aber auch im Vergleich zu qualitativ hochwertigen Zeitschriften kann die *Only!* sich sehen lassen. Mich ärgert immer wieder, wenn z.B. bei Treibern und anderer Software, ist keine nährere Information angegeben, Win gemeint ist. Ich hatte für einige Zeit ein Abonnement der Zeitschrift *OS/2 Inside*, und meine Zufriedenheit mit ihr war durchwachsen. Man führte z.B. Vergleiche mit Compilern durch, aber wie man sich konkret mit einfachen Mitteln eine Entwicklungsumgebung zusammenbastelt, darüber fand ich nichts. Und da bin ich schon beim Thema: Ein Artikel könnte z.B. sich mit dem freien GNU-Compiler gcc beschäftigen. Auf dem Buchmarkt gibt es wirklich gute Bücher zum Thema C und C++, aber alles bezieht sich immer auf Win und Linux. Es ist wirklich schwierig, OS/2-spezifische Informationen und gerade diese auf Deutsch zu bekommen. Man sollte doch alles was mit Entwicklung unter OS/2 zu tun hat, massiv unterstützen. Nur so kann man die Zukunft von OS/2 sichern.

Um noch nochmals auf die *Inside* zurückzukommen, hat mir persönlich der Beitrag über IPF-Programmierung gefallen, der so ausführlich war, das man damit was anfangen konnte. Gerade für zukünftige OS/2-Programmierer könnte die *Only!* also erste Hilfestellungen geben. Sonst könnte man sich zu schnell von dieser Plattform ab- und sich anderen Betriebssystemen zuwenden, wo man es einfacher hat - zumindest was die Informationen betrifft. Ich hoffe, das diese Anregung auf offene Ohren stößt.

Diese Anregung stieß auf offene Ohren. Da wir den Bereich »Programmieren« völlig neu gestalten (siehe Editorial und die Ankündigung auf unserer Homepage), sollte für jeden etwas dabei sein, und auch an OS/2-spezifischen Einführungskursen, etwa in C, sollte es dann nicht mehr mangeln. Was andere Zeitschriften angeht: Wir dürfen natürlich nicht vergleichen - aber Sie...

Anmerkung des Verlages

Auf ein ewig lebendes OS/2

Andreas Hüttner via eMail

Auf diesem Wege möchte ich mich zuerst einmal recht herzlich für die sagenhaft schnelle Zusendung der ersten Ausgabe der *OS/2 Only!* bedanken: Montag nachmittag telefonisch bestellt und Dienstag mittag in der Post gehabt!

Zweitens möchte ich Ihnen sehr zu diesem Werk gratulieren, es ist meines Erachtens das bisher Beste, was ich in Sachen OS/2-Magazinen gelesen habe. Da kam sogar die leider mittlerweile verstorbene *OS/2 Inside* nicht heran.

Auch im Kollegenkreis war das Erstaunen groß, als ich die *OS/2 Only!* auf den

Tisch legte. Jeder wollte sich auch sofort das Magazin bestellen. Gerade OS/2-Anfängern (einige meiner Kollegen) aber auch mir als langjährigen OS/2-Nutzer hat dieses Magazin sehr wohlgetan, reichhaltiges und dennoch leicht verständliches Wissen vermittelt, bzw. verdeutlicht und vertieft.

Ich kann wirklich nur hoffen, daß Sie mit diesem Magazin fortfahren können und nicht aus wirtschaftlichen Gründen *OS/2 Only!* schon nach kurzer Zeit einstellen müssen. Lange Zeit haben die OS/2-Freunde in Deutschland schließlich keine Publikationen zu ihrem geliebten System mehr in den Händen halten können. Schade nur, daß man auf das Magazin eigentlich nur durch Mundpropaganda aufmerksam wird. Leider ist *OS/2 Only!* auch im Zeitschriftenhandel oder Buchhandel nicht so einfach erhältlich.

Sollte sich der Band 1 nicht als Eintagsfliege herausstellen, und die Kontinuität der *OS/2 Only!* gewährleistet sein, so werde ich schnellstmöglich ein Abo abschliessen, um ja keine Ausgabe zu verpassen. Auf ein »ewig lebendes« OS/2 hoffend...

...können wir den Fortbestand des Magazins verkünden. Daß die OS/2 Only! *im Buchhandel nicht so einfach zu bekommen ist, liegt oftmals auch an den Händlern, die sich weigern, Einzelbestellungen für den Kunden durchzuführen - zumal OS/2 nicht gerade ein Thema ist, daß sich auf dem Buchmarkt durch den Handel erstklassig verkaufen läßt. Besonders freut uns natürlich, daß die* OS/2 Only! *auch für Anfänger verständlich ist.*

Anmerkung des Verlages

Zum Artikel ZIP-Drives in Bd.1

Andreas Brogle via eMail

Euer positives Fazit zu *ZIP-Laufwerken* kann ich in keinster Weise teilen. Es ist sicher wünschenswert, daß OS/2 jede erdenkliche Hardware unterstützt. Aus Eurem Beitrag bekomme ich aber den Eindruck, hier wird der Umkehrschluß gezogen: Alles was OS/2 unterstützt, ist gut. Hier wünsche ich mir etwas mehr Kritik. Für mich ist letztlich die Frage, warum sich das *ZIP-Laufwerk* als Wechselmedium durchgesetzt hat, die gleiche wie die, warum sich *Windows* als Betriebssystem durchgesetzt hat.

Ich selbst und zwei meiner Kollegen sind im Besitz eines internen *SCSI-ZIP-Laufwerks*; an der Uni ist ein paralleles Laufwerk an einen *Windows-Rechner* angeschlossen. Das Plastikgehäuse des parallelen Laufwerks als robust zu bezeichnen, hat mich ziemlich verwundert. Ich kann mir nicht vorstellen, daß das Laufwerk einen Sturz vom Tisch überstehen würde. Ich würde mich sogar davor hüten, von der vorgesehenen Möglichkeit Gebrauch zu machen, das Laufwerk hochkannt aufzustellen.

Mein erstes *ZIP-Laufwerk* hat nach einem halben Jahr angefangen, die Medien spiralförmig zu zerkratzen. Einer meiner Kollegen hat innerhalb eines Jahres bereits sein drittes *ZIP-Laufwerk*. Schon zwei Mal ist dort der Schreib/Lesekopf abgebrochen. Unsachgemäße Behandlung kann in allen Fällen ausgeschlossen werden. Ich kann deshalb nur davor warnen, halbwegs wichtige Daten auf dem *ZIP* abzulegen. Die Laufwerke gehen am laufenden Band kaputt und nehmen die Medien gleich mit ins Grab.

Zum Glück gibt es Alternativen, ich kann MODs nur wärmstens empfehlen. Diese setzen meine Kollegen und ich inzwischen schon seit Jahren ein. Defekte Medien sind noch nie vorgekommen, und auch die Laufwerke zeigen keine Mucken. Sie sind wirklich robust und lassen sich zur Reinigung notfalls auch mal öffnen und problemlos wieder zusammenbauen. Optische

Medien sind grundsätzlich viel unempfindlicher als magnetische Medien. Die Medien selbst bieten eine maximale Haltbarkeit. Die Hersteller reden von 30-50 Jahren, während es beim *ZIP* gerademal 10 Jahre sind. Defekte *ZIP-Medien* sind zudem während des Gebrauchs keine Seltenheit. Sogar in der Häufigkeit der Überschreibbarkeit sind MO-Medien Disketten oder *ZIP-Disks* weit überlegen.

Den Preis des *ZIP* als akzeptabel zu bezeichnen, hat mich fast von Stuhl gehauen. Im Gegensatz zum *ZIP* gibt es bei den MO-Medien Konkurrenz und freien Wettbewerb. Folglich betragen die Medienpreise fuer MO-Medien gerademal ein Fünftel im Vergleich zum *ZIP*. Hinzu kommt die Medienkapazität. 100 MByte sind nicht gerade viel, im grunde genommen inzwischen nicht mehr zeitgemäß.

Mich hat das *ZIP* nicht überzeugt und ich komme zu dem Gesamturteil ungenügend. Ich setze sie nur noch ein, weil mich die Uni sozusagen dazu zwingt, genauso wie ich gelegentlich dazu gezwungen werde, Word-Dokumente zu importieren. Beides sind Quasi-Standards, über die ich mich ärgen darf. Wer *ZIP* empfiehlt, muß deshalb auch *Windows* empfehlen. *ZIP* ist nicht nur die schlechteste Variante der Wechselmedien, die Technik ist einfach unbrauchbar, selbst wenn es keine Alternativen geben würde.

Wir haben die hier beschriebenen Probleme mit den ZIPs nicht. Dabei tauschen wir immer wieder die gleichen Medien mit unserem Repro-Studio aus, die Macintosh-Rechner verwenden. Der gegenseitige Datenaustausch klappt problemlos. Daß die ZIP-Laufwerke in der Verarbeitung Mängel aufweisen, erwähnten wir. Was nun das letztliche Urteil angeht: Darüber kann man sich lange streiten. Bezüglich Word stimmen wir zu. Fast ein wenig zu weit geht aber der völlig verkehrte Syllogismus: »Wer ZIP empfiehlt, muß deshalb auch Windows empfehlen.« Bevor wir Windows empfehlen, schließen wir den Betrieb lieber.

Anmerkung des Verlages

Störende Dinge

Ein unzufriedener Leser via eMail

Ich bin leider nicht an dem zweiten Band der *OS/2 Only!* interessiert. Dies liegt jedoch weder am Thema noch am Preis. Konkret stören mich derzeit drei Dinge. Die beiden wichtigsten sind das Layout und die Sprache. Ersteres ist für meine Begriffe sehr unpraktisch (ich kenne die Gründe dafür, aber ich halte das Buchformat trotzdem nicht für akzeptabel), letztere gefällt mir einfach nicht. Für eine Unterhaltung ist diese Sprache in Ordnung, für eine Zeitung fehlt mir jedoch der Feinschliff. Der dritte Punkt betrifft die Web-Seite: Persönlich hätte ich den Inhalt der *OS/2 Only!* auch gerne im Internet - oder genauer ausgedrückt: als HTML Dokumente.

Der Inhalt ist für meine Begriffe ok, auch wenn ich mir etwas aktuellere Berichte gewünscht hätte. Ein Beispiel: Jeder OS/2-Anwender kennt die derzeitigen Probleme, und statt einem Workshop über DLLs hätte man sich meiner Meinung nach besser mit der Portierung von *Linux-Anwendungen* über emx, einem guten Einstiegskurs zu *Java* oder der *Device-Treiberentwicklung* beschäftigt.

Wenn wir aktuelle Texte im Internet publizieren, können wir auf den teuren Gang zum Drucker verzichten.- Und noch einmal: Die OS/2 Only! *ist **keine** Zeitung und auch **kein** herkömmliches Magazin und wird **so** auch nie sein. Unserer Ansicht nach ist das Layout herkömmlicher Zeitschriften zu unpraktisch, und dort scheint der Sprachstil zu leger. Aber was meinen Sie zu diesem Beitrag? Schreiben Sie uns Ihre Meinung.*

Anmerkung des Verlages

Die CD-ROM

Wie Sie aus dem Inhaltsverzeichnis ersehen können, dreht sich in diesem Band der *OS/2 Only!* alles um die **Compact Disk (CD)** und deren Herstellung mit dem eigenen OS/2-System. Sowohl die **CD-Recorder**, die wir für Sie getestet haben, als auch die Software zum Beschreiben der **CD-Rs** werden viel besser verständlich und leichter anzuwenden sein, wenn man mehr über den Datenträger weiß, den man beschreiben und benutzen möchte. Dieses Kapitel dient dazu, Sie mit der »Anatomie« einer *CD-ROM* vertraut zu machen.

CD-ROM?

Seit Markteinführung der *Compact Disks* im Audiobereich erlangte die *CD* rasant eine hohe Bedeutung sowohl in der Musik- als auch in der Computerindustrie. Ohne jeden Zweifel kann man sagen, daß die multimedialen Fähigkeiten der PCs, etwa die Entwicklung von Soundkarten und **MIDI-Schnittstellen** des Rechners durch die *CD-ROM* eine besondere Förderung erfahren haben. Heutzutage ist die *CD* aus der EDV nicht mehr wegzudenken, und während noch vor einigen Jahren das Beschreiben von *CDs* für den Heimanwender völlig unrentabel und auch technisch schwierig durchführbar war, ist es heute kein Problem mehr, die Vorteile der *CDs* auch im kleinen Maßstab voll auszunutzen.

CDs und CD-Rs

Wer CDs beschreiben möchte, verwendet dazu genau genommen eine *CD-Recordable (CD-R)*. Auf einen normalen CD-Rohling kann man nicht schreiben, da *CDs*, wie man sie kennt, gepreßt werden. Wir arbeiten diesen Unterschied im folgenden noch genauer aus, werden uns aber für beide CD-Typen auf den allgemeineren Terminus *CD* einigen, denn auch eine *CD-R* ist eine *CD*. Daneben gelten bis auf einige Unterschiede genau die gleichen Restriktionen, Vor- und Nachteile. Trotz allem sei vor den folgenden Ausführungen auf die Terminologie noch einmal hingewiesen.

Wie der Name schon sagt, ist eine *CD* nur einmal beschreibbar. Anschließend kann von dem Datenträger nur noch gelesen werden (daher der Appendix *ROM = Read Only Memory*). Ein erneutes Beschreiben ist also nicht mehr möglich. Eine Ausnahme stellen die **CD-RWs** (CD-Rewritables) dar, die wiederbeschreibbar sind. Für das Beschreiben und Wiederbeschreiben dieses CD-Typs ist ein spezieller Recorder nötig. Alles zu den **CD-RWs** lesen Sie weiter unten in diesem Kapitel und in den Praxistests der *CD-Recorder* und der *CD-Writer-Software*.

Wie eine CD aufgebaut ist

Eine herkömmliche, gepreßte *CD* besteht in der Regel aus drei Schichten. Die untere Schicht, die man auch Substrat nennt, besteht aus Kunststoff. Auf ihr befindet sich eine ca. 45-105 nm dicke metallene Reflexionsschicht, meist aus Aluminium, die auf den Kunststoff aufgedampft wird. Die CD wird labelseitig durch eine eine weitere Kunststoffschicht abgeschlossen, auf die man z.B. einen Aufdruck anbringen kann. Insgesamt ist die CD damit etwa 1,2 mm dick und besitzt einen Durchmesser von 12 cm.

Eingeschlossen zwischen Substrat und Schutzschicht ist die reflektierende Metallschicht recht gut gegen Umwelteinflüsse geschützt. Trotz allem müssen für die Relexionsschicht Metalle verwendet werden, die mit Feuchtigkeit und Sauerstoff so gut wie gar nicht rea-

gieren. Demnach kommen als Trägerschichten Aluminium, Silber, Gold oder Platin in Frage.

Eine *CD-R* unterscheidet sich von einer herkömmlichen *CD* neben einer unter der obersten Schutzschicht aufgebrachten dünnen Schicht aus UV-undurchlässigem Lack durch eine Aufnahmeschicht, die zwischen Substrat und der Reflexionsschicht liegt. Diese Schicht besteht aus einem organischen Farbstoff und kann durch den Laserstrahl des *CD-Recorders* einmalig verändert werden. Je nach Art dieses Farbstoffes kann die untere Seite einer *CD-R* blau, grün oder blaugrün sein. Die Reflexionsschicht einer *CD-R* besteht meist aus 24Kt. Gold[1]. Dadurch wird der Preis jedoch nicht sonderlich erhöht, da die Schicht wie schon erwähnt äußerst dünn ist. Den Aufbau einer normalen *CD* und einer *CD-R* zeigt *Abbildung 1*.

Auf der Unterseite werden die Daten in einer Spirale abgelegt, die von innen nach außen führt und eine Länge von ca. 6 km hat. Daher werden *CD-ROMs* auch immer von innen nach außen beschrieben. Auf noch unbeschriebenen *CD-Rs* befindet sich bereits eine solche vorgefertigte Spirale. Auf diese Weise läßt sich der Laserstrahl des Recorders beim Schreiben über die Oberfläche der *CD-R* »leiten«, was den Aufbau der Recorderhardware erheblich vereinfacht. Eine Schallplatte beinhaltet die Audiodaten übrigens auch in einer solchen Spirale, die jedoch außen beginnt. Während auf einer Schallplatte

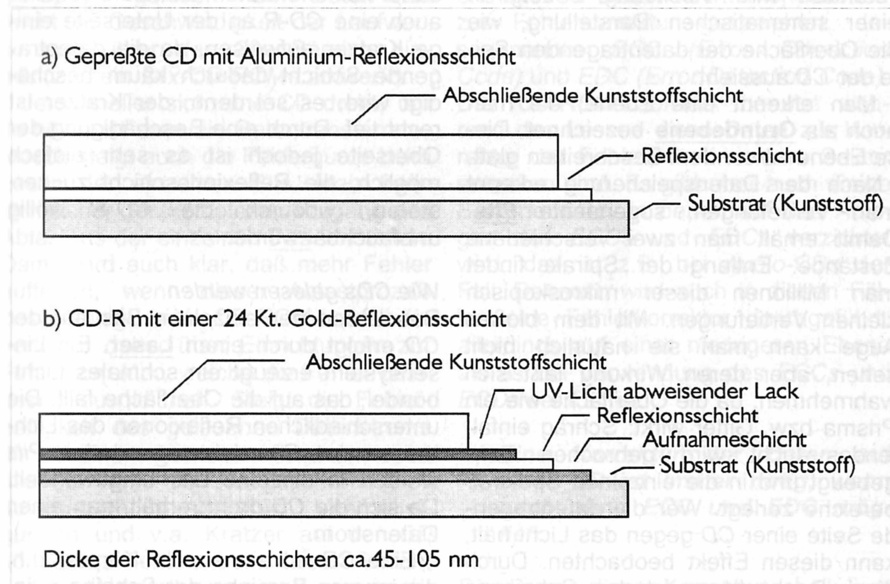


Abb. 2: Der Aufbau einer handelsüblichen, gepreßten CD mit Aluminiumreflexionsschicht (a) und einer CD-R mit einer Reflexionsschicht aus 24-karätigem Gold (b). Beide Schichten sind im Mittel nur etwa 70 nm dick.

1 Kt. ist die Abkürzung für Karat, einer Gewichtseinheit der Juweliere, Gold- und Silberschmiede. Die Karatzahl gibt im Falle von Metallen den Feinheitsgrad einer Legierung an. 24 Kt. Gold stehen dabei für Feingold, also »reines« Gold.

Audiodaten allerdings in analoger Form festgehalten sind, werden Daten auf einer *CD* in digitaler Form abgelegt, d.h. als Bits. Insgesamt können bis zu 650 MByte auf einer *CD* gespeichert werden[2]. Wie Sie aus *Bd.1* von der Betrachtung der CPU wissen, kann ein Bit den Wert 0 oder 1 annehmen. Bei der *CD* werden diese Zustände gespeichert, indem die datentragende Schicht verändert wird. *Abbildung* 2 zeigt in einer schematischen Darstellung, wie die Oberfläche der datentragenden Seite der *CD* aussieht.

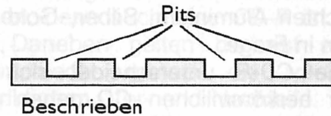


Abb. 2: Schematische Darstellung der datentragenden Oberfläche einer CD. Im unbeschriebenen Zustand ist die Oberfläche glatt. Nach dem Beschreiben sind Vertiefungen (Pits) in der Beschichtung enthalten. So wird eine Codierung der auf der CD befindlichen Daten in Bits möglich.

Man erkennt eine Ebene, die man auch als **Grundebene** bezeichnet. Diese Ebene ist vor dem Beschreiben glatt. Nach der Datenspeicherung erkennt man Vertiefungen, sogenannte **Pits**. Damit erhält man zwei verschiedene Zustände. Entlang der Spirale findet man Millionen dieser mikroskopisch kleinen Vertiefungen. Mit dem bloßen Auge kann man sie natürlich nicht sehen, aber deren Wirkung läßt sich wahrnehmen, da die Oberfläche wie ein Prisma bzw. Gitter wirkt: Schräg einfallendes Licht wird gebrochen bzw. gebeugt und in die einzelnen Spektralbereiche zerlegt. Wer die datentragende Seite einer *CD* gegen das Licht hält, kann diesen Effekt beobachten. Durch diese Beobachtung ist das Geheimnis der *CD* auch schon entzaubert: Durch die Vertiefungen wird die Reflexionseigenschaft der datentragenden Oberfläche verändert. Damit wird die Rekonstruktion der auf der *CD* befindlichen Daten ermöglicht.

Bei einer *CD-R* wird die Aufnahmeschicht durch den Laser entsprechend verändert, um die einzelnen Pits zu erzeugen. Die Relfexionsschicht trägt daher keinerlei Daten. Demzufolge kann auch eine *CD-R* an der Unterseite einige Kratzer aufweisen, da die datentragende Schicht dadurch kaum beschädigt wird, es sei denn, der Kratzer ist recht tief. Durch eine Beschädigung der Oberseite jedoch ist es sehr einfach möglich, die Reflexionsschicht zu zerstören, wodurch die *CD-R* völlig unbrauchbar würde.

Wie CDs gelesen werden

Das Lesen der einzelnen Bits auf der *CD* erfolgt durch einen **Laser**. Ein Linsensystem erzeugt ein schmales Lichtbündel, das auf die Oberfläche fällt. Die unterschiedlichen Reflexionen des Lichtes an der *Grundebene* und den *Pits* werden in einzelne Bits umgewandelt. Da sich die *CD* dreht, erhält man einen Datenstrom.

Eine CD ist ein starrer Körper, d.h. die inneren Bereiche der Scheibe rotieren langsamer als die äußeren. In einer Sekunde werden ca. 1,3m der 6 km langen Datenspirale gelesen, womit sich

2 Das entspricht ca. 74 Minuten Musik, wobei die Tondaten mit 44.1 KHz in Stereo gesampelt (digitalisiert werden).

Rotationsgeschwindigkeiten von 200 U/min in den inneren Bereichen bis 500 U/min in den äußeren Bereichen der *CD* ergeben. Dabei werden pro Sekunde 150 KByte gelesen. Bei dieser Datentransferrate bezeichnet man das Laufwerk, in dem sich die *CD-ROM* befindet, als *Single Speed-CD-ROM-Laufwerk* (das sind z.B. alle *Audio-CD-Player*). Man kennzeichnet das Laufwerk auch kurz mit 1x (1-fach). Natürlich kann man die *CD* auch schneller rotieren lassen und damit den Datenstrom auf mehr Byte pro Sekunde erhöhen. Bei einer Datentransferrate von 300 KByte/s spricht man von einem *Double Speed (2x)-Laufwerk*, bei 600 KByte/s von einem *Quad Speed (4x)-Laufwerk* usw. *Audio-CDs* werden immer mit *Single-Speed* gelesen. Bei *Daten-CDs* reicht die Geschwindigkeit der Laufwerke auch über den 20er-Bereich hinaus (z.B. ein *24x-CD-ROM-Laufwerk*).

Je schneller sich eine CD dreht, um so höher ist zwar die Datentransferrate; jedoch steigt auch die Fehlerquote beim Lesen der einzelnen Bits. Man kann sich sicher den komplexen Vorgang des Abtastens der einzelnen **Pits** vorstellen. Damit wird auch klar, daß mehr Fehler auftreten, wenn dieser Abtastprozeß beschleunigt wird. *CD-ROM-Laufwerke* verfügen daher über Einrichtungen zur Fehlerkorrektur, welche die Fehler beim Lesen ausgleichen. Steigt die Fehlerhaftigkeit des gelesenen Datenstroms bis zu einem gewissen Grad an, setzt das Laufwerk die Lesegeschwindigkeit automatisch herab. Auch Verunreinigungen und v.a. Kratzer auf der *CD* können zu Fehlern beim Lesen führen, und der Herstellungsprozeß verursacht in jedem Fall ebenfalls Fehler (ohne daß die *CD* dabei unbrauchbar würde). Somit wird ein *CD-ROM-Laufwerk* selten mit der maximal möglichen Geschwindigkeit arbeiten. Ein solches Laufwerk daher nach der höchsten vom Hersteller angegebenen Lesegeschwindigkeit zu kaufen, führt zu dem gleichen Irrtum, dem Anwender verfallen, wenn Sie meinen, ein schneller Prozessor könnte das Fehlen von Hauptspeicher kompensieren. Wer ein leistungsfähiges *CD-ROM-System* erwerben möchte, investiere lieber in ein besseres Subsystem (**SCSI** statt **IDE**). Die physikalische Beschaffenheit der *CD* setzt der Hardware einfach Grenzen.

Bezüglich der Fehlerkorrektur ist noch einiges zu sagen. Wir werden bei der Betrachtung der *Trackmodi* erfahren, daß die Daten auf einer *CD* in Sektoren abgelegt werden, die ein unterschiedliches Format haben können. Meistens werden in einem Sektor für die darin enthaltenen Daten noch Informationen zur Fehlerkorrektur gespeichert, der sogenannte *ECC (Error Correction Code)* und *EDC (Error Detection Code)*. Das *CD-ROM-Laufwerk* benutzt während des Lesens diese Daten zur Korrektur der Fehler, die in einem Sektor enthalten sind. Es gibt aber auch einige Sektorformate, in denen auf das Ablegen des *ECCs* und *EDCs* verzichtet wird, das ist z.B. bei *Audio-CDs* der Fall. Dennoch wird auch in diesen Fällen eine Fehlerkorrektur durchgeführt, allerdings auf einer niedrigeren Ebene als es bei Verwendung des *ECCs* und *ECDs* der Fall ist.

! ***Eine Korrektur der Daten auf einer CD ist immer nötig, wenn auch ohne ECC und EDC möglich.***

Das Prinzip dieser Korrekturtechnik liegt in der Art und Weise, wie Daten auf einer *CD* enthalten sind. Diese werden mit dem sog. *CIRC (Cross-Interleaved Reed-Solomon Code)* gespeichert, genauer: codiert. Beim Lesen werden

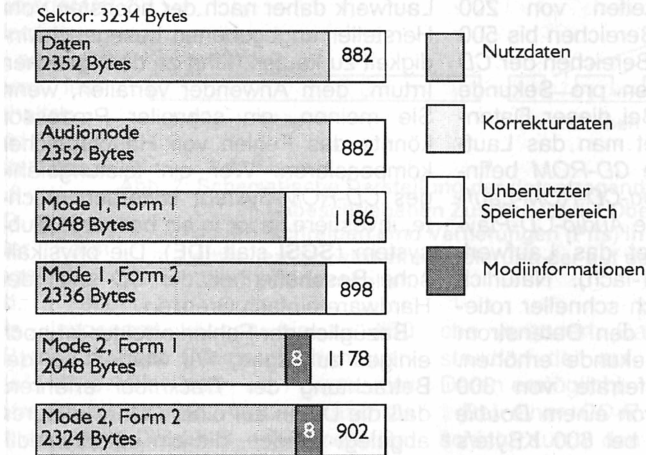


Abb. 3: Die verschiedenen Trackmodi und ihr Aufbau

die Daten durch das Laufwerk (oder den CD-Player) decodiert, wobei zwei verschiedene Ebenen der Korrektur existieren:

1. C1, eine Methode, die eine Fehlerkorrektur auf Bit-Ebene ermöglicht.
2. C2, eine Methode, die Fehlerkorrekturen auf Byte-Ebene gestattet, wobei einzelne Bytes zu 24 Byte großen Frames und diese zu 98 Frame großen Datenblöcken zusammengefaßt werden.

Mit diesen Codierungsverfahren können die meisten Fehler korrigiert werden. Steigt die Fehlerrate jedoch weiter an (weil die *CD* z.B. sehr verschmutzt oder gar verkratzt ist), so beginnt das Laufwerk die fehlerhaften Bereiche des Datenstroms durch interpolierte Werte zu ersetzen. Auf diese Weise wird man beim Anhören von *Audio-CDs* vor störenden Knackgeräuschen zumeist verschont. Allerdings erreicht die Methode der Interpolation, wie man sich leicht vorstellen kann, rasch ihre Grenze, womit der Sinn zusätzlichen *ECCs* und *EDCs* klar wird.

Die mittlere Zugriffszeit eines *CD-ROM-Laufwerkes* ist übrigens bedeutend größer als die von Festplatten. Während bei letzteren die Zugriffszeiten im Millisekundenbereich liegen, verfügen *CD-ROM-Laufwerke* über mittlere Zugriffszeiten im Bereich einiger Zehntelsekunden (in der Regel 0,2 bis 0,7 Sekunden). Die effektive Zugriffszeit ist trotz schneller Hardware allerdings auch durch den Aufbau einer *CD* begrenzt. Um zu verstehen, wie Daten auf einer *CD* abgelegt und verwaltet werden, ist ein genauere Betrachtung der Struktur einer *CD-ROM* nötig.

Über Tracks, TOCs und Sessions

Die physikalische Struktur einer *CD* ist nun hinreichend bekannt. Betrachten wir die *CD* jetzt vom Standpunkt der Datenspeicherung und -Verwaltung.

Die Daten einer *CD* (egal ob Audiodaten oder Dateien) werden in sog. **Tracks** (Spuren) gespeichert. Diese Spuren können eine variable Länge haben. Die Tracks selbst bestehen wiederum aus einzelnen **Sektoren** mit einer invariablen Länge von 3234 Bytes. Auf einer *CD* können maximal 99 Tracks vorhanden sein. Die Daten werden also sektorweise innerhalb eines Tracks geschrieben und gelesen.

Pro Sektor können 2352 Bytes Nutzdaten gespeichert werden. Die verbleibenden 882 Byte dienen zur Aufnahme von Informationen für die Fehlerkorrektur. Die Sektoren können intern unterschiedlich organisiert sein. Es gibt fünf sog. Trackmodi, welche diese Organisation festlegen (siehe auch *Abb. 3*):

1. *Audiomode*

   In diesem Modus werden die gesamten 2352 Bytes eines Sektors mit Audiodaten belegt. Auf Informationen für eine Fehlerkorrektur wird verzichtet. Sämtliche Fehler auf der CD werden also auch wiedergegeben, liegen allerdings außerhalb des menschlichen Hörbereiches, so daß sie von uns nicht wahrgenommen werden können.

2. *Mode 1, Form 1*

   Dieser Modus ist das Standardformat für Daten-CDs. Pro Sektor werden 2048 Byte mit Daten beschrieben. Der restliche Bereich wird zur Speicherung von Informationen für die Fehlerkorrektur verwendet.

3. *Mode 1, Form 2*

   Ähnlich wie im *Audiomode* wird hier auf das Speichern von Daten zur Fehlerkorrektur verzichtet. Ein Sektor kann daher 2336 Bytes aufnehmen. Dieses Format ist ungebräuchlich.

4. *Mode 2, Form 1*

   Das Format ähnelt dem Trackmodus Mode 1, Form 1. Der Modus Mode 2, Form 1 wird v.a. für *Kodak Photo-CDs* verwendet. Hierbei werden 8 nicht verwendete Bytes des Speicherbereiches zur Fehlerkorrektur zum Ablegen von Modiinformationen verwendet. Damit wird es möglich, zwischen verschiedenen auf der CD enthaltenen Sessions und unterschiedlichen Trackmodi umschalten zu können. Die Herstellung von Multisession-CDs und das Speichern unterschiedlicher Sektorformate innerhalb eines Tracks werden auf die Weise ermöglicht. Die Speicherkapazität bei Verwendung dieses Trackmodus pro Sektor beträgt 2048 Byte.

5. *Mode 2, Form 2*

   Dieser Modus ähnelt Mode 1, Form 2. Das Datenformat entspricht dem des Mode 2, Form 1, auf eine Fehlerkorrektur wird jedoch weitgehend verzichtet, so daß insgesamt 2324 Bytes Nutzdaten in einem Sektor gespeichert werden können. *CDs*, die mit diesem Format geschrieben sind, können von älteren Laufwerken, die nicht *Photo-CD*-kompatibel sind, nicht gelesen werden.

Mit diesen Formaten kennen Sie nun den Aufbau der einzelnen *Sektoren* innerhalb eines *Tracks*. Der *Track* selbst kann aus beliebig vielen solcher *Sektoren* bestehen.

Eine CD beinhaltet mindestens einen *Track*. Eine Vorschrift, wieviele *Spuren* eine CD desweiteren enthalten muß, gibt es nicht, nur die Höchstzahl von 99 kann nicht überschritten werden[3]. Wichtig ist zu beachten, daß eine Datei oder ein Lied nur innerhalb einer Spur abgespeichert wird. Es ist also nicht üblich, eine Datei oder ein Musikstück auf mehrere Tracks zu verteilen. Das rührt

---

3 Die Tracknummern werden BCD-codiert gezählt, also als zweistellige Dezimalzahl beginnend bei 01. Daher kann die maximale Anzahl von 99 Tracks nicht überschritten werden.

daher, daß die Verzeichnisinformationen nicht in einem gesonderten Verzeichnisbaum, sondern im *Track* selbst enthalten sind: Jeder *Track* führt sein eigenes Verzeichnis (*Directory*) der Informationen, die nur er enthält. Dabei gilt:

- Beliebig viele Dateien können innerhalb eines *Tracks* (Datentrack) abgelegt werden.
- Ein Musikstück kann nur in einem eigenen *Track* (Audiotrack) abgelegt werden (ein Musikstück pro Spur).

Im übrigen folgt einem abgeschlossenen Track nicht sofort der nächste: Zwischen jedem Track befindet sich ein nicht bespielter Bereich, ein sog. **Pregab**. Sie kennen ihn als Pause von ca. 2 Sekunden Länge auf Ihren Audio-CDs. Dieser *Pregab* befindet sich sowohl auf Audio- als auch auf Daten-CDs.

TOCs
In einem Track befinden sich Informationen über die einzelnen Positionen der Sektoren innerhalb des Tracks und wozu sie gehören; ferner allgemeine Systeminformationen über den Track selbst.

Da bis zu 99 Tracks auf einer *CD* enthalten sein können, ist ein weiteres, übergeordnetes Verzeichnis nötig, die sogenannte **Table Of Contents (TOC)**. Die *TOC* speichert übrigens keine *Directory*-Informationen, sondern lediglich die Anzahl der *Spuren* und deren Anfangspositionen auf der CD. Ein solches Verzeichnis wird zu Beginn einer jeden *Session* neu angelegt. Genau gesagt ist die *TOC* Bestandteil des sog. *Lead-In*, mit dem eine *Session* beginnt. Was eine Session ist, erklärt der nächste Abschnitt.

Lead-In, Lead-Out und Sessions
Der Anfang der kilometerlangen Datenschleife auf der *CD* beginnt mit einem Datenblock, der grundsätzliche Informationen über die *CD* beinhaltet, etwa der beim Beschreiben benutzte Trackmodus. Dieser Informationsblock wird **Lead-In** genannt.

Einen abgeschlossenen Aufzeichnungsvorgang bezeichnet man als **Session** (Sitzung). Eine *Session* beginnt mit einem *Lead-In*, der auch die *TOC* enthält, dem beliebig viele, jedoch nicht mehr als 99 Tracks folgen[4].

Die *Session* wird mit einem *Lead-Out* abgeschlossen, dessen Inhalt weniger relevant als der des *Lead-Ins* ist.

Im Normalfall beinhaltet eine *CD* nur eine *Session*, d.h. der Datenträger verfügt nur über einen *Lead-In*, der die *Session* einleitet und einen **Lead-Out**, der sie abschließt. Diese Informationsblöcke *müssen* vorhanden sein, wenn man die *CD* in einem *CD-ROM-Laufwerk* lesen möchte.

Während man eine *CD-R* noch beschreibt, ist der Datenträger noch nicht »abgeschlossen«, d.h. *Lead-In*

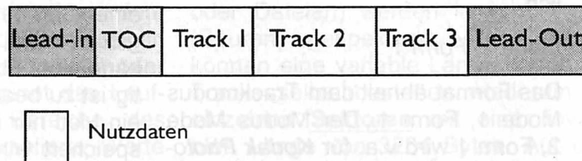


Abb. 4: Abfolge von Lead-In, TOC, Tracks und Lead-Out.

4 Die Tracks werden sessionübergreifend gezählt. Befinden sich in einer Session 40 Tracks, kann die zweite demnach höchstens 59 beinhalten.

und *Lead-Out* sind in diesem Stadium noch nicht auf dem Datenträger vorhanden. Um aber die bereits auf die *CD-R* geschriebenen Trackpositionen zu speichern, wird die *TOC* in einen speziellen Speicherbereich, der *PMA (Program Memory Area)* geschrieben, auf den nur CD-Recorder zugreifen können. Daher ist man nicht gezwungen, eine *Session* abzuschließen, sondern kann das Beschreiben der *CD-R* später fortsetzen. In normalen *CD-ROM-Laufwerken* ist die *CD* jedoch noch nicht lesbar. Erst nach dem Abschluß der *Session* wird der *Lead-In* geschrieben und in ihn die *TOC*. Wird eine *CD-R* abgeschlossen, werden für den *Lead-In* und *Lead-Out* etwa 12-20 MByte der Speicherkapazität beansprucht.

Natürlich kann der Speicherplatz auf einer *CD* nur einmal beschrieben werden; ist aber noch Speicherkapazität vorhanden, kann man eine neue *Session* beginnen, die hinter der ersten angefügt wird. Bei der Herstellung einer solchen *Multisession-CD* muß man die neue *Session* bereits erstellen, wenn die erste abgeschlossen wird. Das ist deswegen so, weil der erste *Lead-In* einen Verweis auf den zweiten enthält, so daß das *CD-ROM-Laufwerk* die *TOCs* der einzelnen *Sessions* als Kontinuum lesen kann. Weil jeder Speicherbereich nur einmal beschrieben werden kann, muß der Verweis auf die zweite *Session* bereits bei Abschluß der ersten in den *Lead-In* aufgenommen werden. Würde die *CD-R* nach Schreiben der ersten *Session* ganz abgeschlossen werden, was man *versiegeln* nennt, so wäre ein späteres Anfügen einer neuen *Session* nicht mehr möglich. Die zweite *Session* kann dann einen Verweis auf die dritte, die dritte einen auf die vierte usw. tragen. Ältere *CD-ROM-Laufwerke* und *Audio-CD-Player* können jedoch nicht von *Lead-In* zu *Lead-In* springen, um die *CD* vollständig zu lesen. Daher erkennen Sie nur die erste *Session* einer *Multisession-CD*.

Die **Multisessiontechnologie** erlaubt eine höhere Flexibilität bei der Fertigung eigener *CDs*. So ist man nicht dazu gezwungen, eine *CD* abzuschließen und zu versiegeln, wenn noch 100 MByte auf dem Rohling frei sind. Man schließt die erste *Session* einfach ab und beginnt eine neue. Allerdings muß das *CD-ROM-Laufwerk,* mit dem die *CD* gelesen werden soll, multisessionfähig sein. Eine Einschränkung, die heutzutage jedoch kaum mehr ins Gewicht fallen dürfte. Übrigens: Eine *CD*, die als *Goldmaster* dient, sollte keine *Multisession-CD* sein.

Zu beachten ist außerdem, daß *Tracks* sessionübergreifend numeriert werden, daß also eine CD stets nur 99 Spuren haben kann, gleich wieviele *Sessions* sich auf ihr befinden. Wie eine *CD* mit *Lead-In*, *Lead-Out*, *TOCs* und *Tracks* aussieht, zeigt *Abbildung 4*.

Über Normen und Namen

Nichts geht ohne Norm, und auch die Datenformate von CDs sind standardisiert. Die **ISO (International Standards Organisation)** legte in der Norm 9660 fest, wie Daten auf CDs gespeichert werden. Was dieser Standard beinhaltet, haben Sie in den vorangegangenen Abschnitten gelesen. Der ISO 9660-Standard bietet ferner zum Abspeichern von Verzeichnisinformationen 3 Stufen, die zur Vergabe von Dateinamen auf einer CD gültig sein können:

1. Level 1

   schreibt das 8.3-Format für Dateinamen vor. Dieses Format kennen Sie von **FAT**. Lange Dateinamen werden einfach auf 8.3 während des Beschreibens der *CD* gekürzt.

2. Level 2

   gestattet Dateinamen mit einer Länge bis zu 31 Zeichen. Der Name darf einen Punkt enthalten.

3. Level 3

   erlaubt wie Level 2 Dateinamen mit einer Länge bis zu 31 Zeichen, gestattet zusätzlich aber beliebig viele Punkte in der Dateibezeichnung.

Für alle drei Level gilt, daß alle Buchstaben großgeschrieben werden, Umlaute und Sonderzeichen durch ~ und Leerzeichen durch _ ersetzt werden.

Neben diesen drei Stufen des ISO 9660-Standards existieren noch zwei Erweiterungen, welche die Einschränkungen der Norm aufheben:

1. *Rockridge Extensions*

   Ein Standard, der die ISO 9660-Norm durch Spezifika des *UNIX*-Dateisystems erweitert. Sowohl auf diesem System verwendete Dateiattribute als auch lange Dateinamen mit Leer- und Sonderzeichen, beliebig vielen Punkten usw. können in diesem Format auf die CD geschrieben werden.

2. *Joliet*

   Der MS-Extra-Windows-CD-Dateisystem-Standard. *Joliet* wird unter *Win9x* und *Windows NT* benutzt. Auch hier können lange Dateinamen gespeichert werden, die bis zu 64 Zeichen lang sein dürfen. Sonderzeichen sind erlaubt.

OS/2 unterstützt von Haus aus den ISO 9660-Standard und Dateinamen nach Level 2. Dies reicht für eine CD auch aus und garantiert, daß die Verzeichnisinformationen auf jedem Betriebssystem ohne Probleme lesbar sind (bis auf MS-DOS, aber das dürfte kaum interessieren). Seit dem FixPack 36 für *Warp 3* und 6 für *Warp 4* unterstützt OS/2 auch den *Joliet*-Standard. Wie er genutzt werden kann, ist im *Software-Kapitel* erklärt, wenn wir *CD-Writer-Software für OS/2* betrachten.

Photo-CD und XA

Vielleicht können Sie sich noch an die Markteinführung der *Kodak Photo-CDs* erinnern. Man konnte sich zum ersten Mal seine Photos direkt vom Film auf die *CD* kopieren lassen. Dann bekam man die *CD* zurück und konnte sie erneut einschicken. Die *Photo-CD* ist also, sofern sie mehr als einmal beschrieben wird, eine Multisession-CD. Als *Trackmodus* kommt *Mode 2, Form 1* zur Anwendung.

Die *Photo-CD* ist schon recht lange auf dem Markt und wurde schnell von den Herstellern der *CD-ROM-Laufwerke* unterstützt. Viele, selbst sehr günstige *CD-ROM-Laufwerke* sind *Photo-CD*- und daher auch multisessionfähig. Neben den eigentlichen Bilddateien enthält die *Photo-CD* noch Steuerinformationen für CDI-Geräte, was für uns jedoch uninteressant ist.

XA ist ein Standard, der zur gleichzeitigen Wiedergabe von Bild- und Tondateien entwickelt wurde. Ein simultanes Lesen beider Datentypen ist v.a. für auf CD gespeicherte Filme notwendig. Erreicht wird es, indem innerhalb der einzelnen Tracks mal Bild- mal Audiosektoren aneinandergereiht werden. *XA-CDs* werden daher im Trackmodus Mode 2, Form 1 oder Form 2 beschrieben.

CDs beschreiben

Sie wissen nun genau, wie eine CD aufgebaut ist, und wie von ihr gelesen wird.

Aber wie wird sie beschrieben und hergestellt?

Die *CD-ROMs*, mit denen man es normalerweise zu tun hat, werden gepreßt (ähnlich wie Schallplatten), nicht gebrannt wie *CD-Rs*. Die Rohlinge haben wir bereits am Anfang dieses Abschnittes beschrieben. Zur Fertigung wird ein sogenannter *Glasmaster* erstellt, der für den Preßvorgang benötigt wird. Abbild für dieses Original ist der bereits erwähnte *Goldmaster*. Die Herstellung des *Goldmasters* bezeichnet man auch als *Premastering*.

Prinzipiell kann man zur Herstellung der Preßvorlage Daten auch auf Medien wie DAT-Streamerkassetten zur Verfügung stellen. Weitaus praktischer ist es jedoch, als *Goldmaster* eine **CD-R** herzustellen, da sie direkt zur Fertigung des *Glasmasters* verwendet werden kann. Da wir uns in dieser Ausgabe schwerpunktmäßig mit der Handhabung von *CD-Rs* befassen, betrachten wir abschließend noch einmal zusammenfassend die Besonderheiten dieses Datenträgers unter Erwähnung weiterer Kleinigkeiten.

CD-Rs und CD-RWs

Eine *CD-R* gehört zu den sog. *WORM-Medien (Write Once Read Multiple Medien)* und ist eine einmalig beschreibbare *CD*, die sich von einem normalen »Silberling« durch eine organische Schicht zwischen Substrat und Reflexionsschicht unterscheidet, die zur Aufnahme der Daten dient. Wie Sie wissen, werden die einzelnen Bits durch einen Laser von der *CD* abgetastet, was möglich ist, da deren Oberfläche unterschiedliche Reflexionseigenschaften aufweist. Um nun auf einem *CD-R-Rohling* die erforderlichen *Pits* zu erzeugen, muß die Oberfläche ihre Reflexionseigenschaften einmalig verändern können. Die Veränderung der Aufnahmeschicht wird thermisch herbeigeführt, und zwar ebenfalls mit einem Laser.

Die Beschichtung einer *CD-R* ist natürlich so widerstandsfähig, daß der Laser normaler *CD-ROM-Laufwerke* zum Abtasten die Oberflächeneigenschaften nicht zerstört. Folgerichtig muß der Laserstrahl zum Schreiben eine höhere Energie haben. Wie die gepreßte *CD* verfügt die selbstbeschriebene *CD-R* über vielerlei Vorzüge:

- Hohe Datensicherheit und Langlebigkeit des Datenträgers,
- einfache Handhabung beim Lesen und Beschreiben und berührungsfreies (optisches) Abstasten der Datenträgeroberfläche,
- simple Austauschbarkeit von Daten von einem auf ein anderes Betriebssystem,
- akzeptable Zugriffsgeschwindigkeiten bei der Suche nach bestimmten Dateien sowie
- extrem niedrige Datenträgerpreise (DM 2,80 bis 5,80) in Relation zur hohen Kapazität von ca. 650 MByte.

Die Lebensdauer eines Datenträgers bestimmt zu großen Teilen den Grad der Datensicherheit des Mediums. *CDs* haben i.a. eine Lebensdauer von bis zu 50 Jahren, wobei herkömmliche *CDs* wegen ihrer Aluminiumbeschichtung eine geringere Lebenszeit aufweisen.

Allerdings ist die Datensicherheit einer *CD-R* letztlich nicht ganz so hoch wie die einer gepreßten *CD*. Die Aufnahmeschicht der *CD-R* ist weitaus empfindlicher als die Beschichtungen der normalen *CD* und nach wie vor empfindlich gegen thermische Einwirkungen. Sonnenlicht etwa kann eine *CD-R* völlig unbrauchbar machen. Man sollte Sie

also von starken Lichtquellen fernhalten, da die Oberfläche ansonsten zerstört werden kann. Auch Verschmutzungen, Kratzer und Umwelteinflüsse greifen eine *CD-R* mehr an als normale *CDs*. Man sollte also vorsichtiger mit diesen Datenträgern umgehen.

Ferner hat die besondere Beschichtung der *CD-R* etwas andere Reflexionseigenschaften als die Metalloberfläche der gepreßten Silberlinge. Demzufolge können beim Lesen mehr Fehler als gewöhnlich auftreten. Generell kann man die meisten Schwächen der *CD-R* jedoch durch die Verwendung hochwertiger Rohlinge minimieren. Zumindest Schwierigkeiten beim Lesen traten bei unseren Tests nicht auf. Es ist schwierig anzugeben, welche *CD-R* die beste ist. Tests einschlägiger Fachzeitschriften sollte man nur mit Vorsicht genießen. Die Qualität der erzeugten *CDs* hängt neben dem Rohling auch vom Zusammenspiel mit dem Recorder und letztlich mit den eingesetzten CD-ROM-Laufwerken ab, so daß Tests, die CD-Rohlinge untersuchen, kaum allgemeingültige Resultate erzielen.

Wichtig ist zu beachten, für welche Schreibgeschwindigkeiten sich *CD-Rs* eignen. Übliche *CD-Rs* lassen sich auf jeden Fall mit 4-facher Geschwindigkeit beschreiben. Es gibt auch solche, auf die man mit 6-facher oder höherer Geschwindigkeit aufzeichnen kann.

Was bezüglich der Geschwindigkeit zum Lesen gesagt wurde, gilt in noch bedeutenderem Maße für das Beschreiben von *CD-Rs*: Je geringer die Schreibgeschwindigkeit ist, desto höher ist die Qualität der fertigen *CD*. Wenn Sie also *CDs* als Vorlage für die Serienproduktion erstellen, stellen Sie sicher, daß der *CD-Recorder* mit höchstens 2-facher Geschwindigkeit schreibt. Eine mit 3- oder 4-facher Schreibgeschwindigkeit gefertigte *CD* eignet sich bereits nicht mehr als *Goldmaster*. Mit geringeren Schreibgeschwindigkeiten stellen Sie außerdem sicher, daß die *CD* von weniger guten *CD-ROM-Laufwerken* problemlos gelesen werden kann.

Eine **CD-RW** (Compact Disk Rewritable) hebt den Nachteil der *CD-R* auf. Wie gesagt kann ein Bereich der *CD-R* nur ein einziges Mal beschrieben werden. Wollte man versuchen, auf einen Bereich zu schreiben, der bereits Daten enthält, würde man die *CD-R* zerstören. Anders bei der *CD-RW*, die nach dem Beschreiben formatiert und daraufhin neu beschrieben werden kann. Dabei ist dieser Datenträger im Prinzip genauso aufgebaut wie die *CD-R*, besitzt jedoch noch einige zusätzliche, dielektrische Schichten und eine andere Reflexionsschicht.

Eine *CD-RW* läßt sich nicht beliebig oft löschen und wieder beschreiben. Etwa 1.000 Zyklen sind realistisch, selbst wenn einige Hersteller weit mehr (z.B. 10.000 oder gar 100.000) Zyklen für möglich erklären. Die Praxis setzt der Wiederverwendbarkeit des Mediums einfach Grenzen.

CD-RW-Rohlinge sind bedeutend teuerer als *CD-Rs* (DM 21,00 bis 28,00 pro Medium). Der Medienpreis ist hier immer noch günstiger als der von MO-Medien, und die hergestellten *CDs* lassen sich theoretisch (!) wie die *CD-Rs* mit jedem Computersystem lesen. Man braucht zum Beschreiben von *CD-RWs* auf jeden Fall einen speziellen Recorder, da der Aufbau der Beschichtung einer *CD-RW* anders als die einer *CD-R* ist; doch auch das *CD-ROM-Laufwerk* muß für *CD-RWs* geeignet sein, weil das Reflexionsvermögen einer *CD-RW* unter dem der normalen *CD* und *CD-R* liegt. Als absolut sicheres Austauschmedium für Daten zwischen verschiedensten Systemen kommt die *CD-RW* daher (noch) nicht in Betracht. ❑

OS/2-taugliche CD-Recorder

Nach der Vermittlung des nötigen Basiswissens rund um die *CD* möchten wir uns nun mit der Praxis befassen und einige für OS/2 geeignete *CD-Recorder* vorstellen.

Für OS/2 kommen sowohl Geräte für den SCSI-Bus als auch für den IDE-Port in Frage. Wir haben folgende CD-Recorder einer näheren Betrachtung unterzogen:

- TEAC CD-R55SK
- Philips CDD3600
- Sony CDU 948 S
- Plextor PX-R820Ti
- Yamaha CRW-4416S
- HP SuperStore 8100i

Die Geräte von *Philips*, *Yamaha* und *Hewlett Packard* eignen sich auch zur Verarbeitung von *CD-RWs*. Der *HP SuperStore 8100i* ist ein IDE-Gerät[5], alle anderen Recorder sind SCSI-Laufwerke.

Wir haben das IDE-Gerät nur der Vollständigkeit halber in unseren Test aufgenommen, um zu zeigen, daß sich auch diese Laufwerke unter OS/2 betreiben lassen. Wir halten **SCSI** für die beste Lösung in puncto Massenspeichersysteme für OS/2, da dieses Subsystem die CPU bei Datenoperationen so gut wie nicht belastet und damit für ein System, das wie OS/2 mit **preemptivem Multitasking** arbeitet, am besten geeignet ist[6]. Wer jedoch jetzt CD-ROM-Recording betreiben möchte, und kein SCSI-Subsystem hat (auch nicht in Zukunft), dem steht mit OS/2 kein Hindernis zum Beschreiben eigener *CDs* mit IDE im Weg. Alle, die aber an einen Umstieg auf SCSI denken, sollten mit der Investition in den CD-Recorder warten, bis sie einen SCSI-Adapter ihr eigen nennen. Mit SCSI zu arbeiten, macht das CD-Recording zu einem wahren Genuß, weil man die *CD-Rs* ohne Hindernisse im Hintergrund beschreiben und normal mit dem System weiterarbeiten kann. Für eine ausreichende Datentransferrate ist mit SCSI im Gegensatz zu IDE immer gesorgt.

Auf jeden Fall raten wir von CD-Recorden für den Parallelport ab. Anders als bei einem LS-120 Laufwerk oder einem Zip, auf das man ab und an zur Datensicherung zugreift, ist für das Beschreiben von CDs eine ausreichender Datenstrom notwendig. Die Leistung des Parallelports kann diese Bedingung aber nicht erfüllen. Zudem benutzt man Geräte für die parallele Schnittstelle, wenn man auf Mobilität angewiesen ist. Das CD-Recording kommt hierfür ohnehin kaum in Betracht. OS/2 unterstützt diese Geräte (zum Glück) auch nicht.

Allgemeines zur Installation

Zur Installation der SCSI-Geräte gibt es nicht viel zu sagen, da sie wie jedes andere SCSI-Device einfach an den Bus angeschlossen werden. Allerdings gibt es einige Kleinigkeiten, die es zu beachten gilt:

- Geben Sie dem CD-Recorder nach Möglichkeit eine ID direkt nach den

5 Der Hewlett Packard CD-Recorder ist auch für den SCSI-Bus erhältlich.

6 Ähnlich wie bei Graphikkarten, übernimmt ein Prozessor auf dem SCSI-Adapter das Management des Datenbus. Die CPU gibt nur noch entsprechende Befehle an den Adapter weiter. Dadurch wird die CPU wenig belastet, und Datenoperationen können im Hintergrund ausgeführt werden.

angeschlossenen Festplatten (in der Regel ID 4-2).

- Gestattet der CD-Recorder nur die Belegung von zwei oder drei IDs, wählen Sie in diesem Fall diejenige der höheren Priorität.
- Dem CD-ROM-Laufwerk in Ihrem System geben Sie die nächste ID nach dem Recorder.
- Alle getesteten CD-Recorder sind werkseitig terminiert. Entscheiden Sie anhand Ihrer Bus-Konfiguration, ob der Recorder terminiert werden muß, und schlagen Sie im Abschnitt der Einzelvorstellung der Geräte nach, wie die Terminierung der Laufwerke vorgenommen wird.

Die oben beschriebene Wahl der ID dient dazu, dem CD-Recorder eine höhere Priorität auf dem SCSI-Bus zu geben. Für simultane Zugriffe mehrerer Geräte auf den Bus sollte der CD-Recorder wegen der z.T. kritischen Situationen bei Datentransfers in stark ausgelasteten Systemen sicherheitshalber gegenüber »unwichtigeren« Geräten wie externen Speichersystemen oder CD-ROM-Laufwerken höherprivilegiert sein, sofern Sie mehr mit Festplatte und Recorder arbeiten.

IDE weist als einfacheres System nur eine sehr primitive Hierarchie unter den Laufwerken auf. Dieser Standard erlaubt grundsätzlich nur zwei Geräte am Port. Bei EIDE gibt es zwei Ports, die unabhängig voneinander arbeiten. Eines der Geräte an jedem IDE-Port ist der Master, das andere der Slave. Wie es der Terminus vermuten läßt, ist auch die Priorität verteilt. Beachten Sie daher:

- Konfigurieren Sie nach Möglichkeit den IDE-CD-Recorder als Master am zweiten IDE-Kanal, und das CD-ROM als Slave am zweiten Kanal.
- Bei älteren **VLB-Systemen** betreiben Sie den Recorder als Slave zusammen mit der Festplatte am ersten IDE-Kanal. Der Grund hierfür liegt darin, daß der zweite IDE-Kanal nicht über den 32-Bit Local Bus, sondern über den 16-Bit ISA Bus betrieben wird. Die Leistung des zweiten IDE-Kanals bei Local-Bus-Systemen ist daher geringer.

Die hier dargestellten Tips dienen dazu, den Recordern einen möglichst guten Platz in puncto Performance im System zu verschaffen, um eine optimale Datenübertragung von anderen Quellen, wie Festplatten oder CD-ROM-Laufwerken, zum Recorder zu gewährleisten. Das ist wichtig, damit der Recorder ohne Unterbrechungen auf die CD schreiben kann.

Im übrigen müssen Besitzer sogenannter »langsamer Rechner«, die z.B. mit einem hochgetakteten 486er oder alten Pentium ausgerüstet sind, nicht auf das CD-ROM-Recording verzichten, zumindest nicht mit SCSI. Wir haben alle SCSI-Geräte ebenfalls in einem VLB-System mit einem 486 DX/2-66, 32 MByte RAM und Adaptec 2842A-VL SCSI-Adapter auf dem VLB-Bus getestet und konnten zügig und ohne Schwierigkeiten CD-Rs und CD-RWs beschreiben. Ein Beispiel dafür, was ein »alter« Prozessor mit dem richtigen Massenspeichersystem und ausreichendem Hauptspeicher unter OS/2 leisten kann.

Rahmenbedingungen für den Test

Uns interessierte bei unserem Test v.a. das Preis-Leistungsverhältnis: Welches Laufwerk bietet zu einem akzeptablen Preis hohe Funktionalität und Zuverlässigkeit? Die Laufwerke, die eine Emp-

fehlung wert sind, sollten natürlich auch in einem finanziell vertretbaren Rahmen liegen, v.a. wenn man bedenkt, daß es auch wichtigere Baugruppen in einem Rechner gibt als ein CD-Recorder (z.B. schnelle Festplatten, RAM und eine gute Graphikkarte).

Die Testmaschinen, mit denen wir die Recorder unter die Lupe nahmen, waren neben dem bereits erwähnten 486er auch ein System mit einem IBM/Cyrix P166+, einem Adaptec 2940 SCSI-Adapter, Onboard-EIDE-Controller und entsprechendem Festspeicherausbau mit einer 4,5 GByte IBM-SCSI- und einer 1,2 GByte EIDE-Festplatte von Seagate. Mit diesem System testeten wir auch den CD-Recorder von *Hewlett Packard*. Den 486er erleichterten wir auch seiner SCSI-Festplatte und ließen das System nur mit seiner 850 MByte IBM-IDE-Festplatte und dem Recorder am SCSI-Bus arbeiten, um die Tauglichkeit der CD-Recorder auf einem System zu testen, bei dem das Festplattensystem nur auf IDE basiert. Wir veränderten die Konfiguration in beiden Systemen auch so, daß IDE gar nicht mehr zum Einsatz kam: Ohne Zweifel die beste Lösung. Alle Recorder testeten wir mit diesen Systemen und den jeweiligen beschriebenen Konfigurationen, um ein möglichst umfassendes Gesamturteil erstellen zu können[7]. Wichtig war uns die gemeinsame Arbeit des Recorders mit IDE v.a. deswegen, da diejenigen, die auf SCSI umgestiegen sind, ihre »alten« IDE-Festplatten sicher nicht fortwerfen; zum anderen interessierte uns natürlich auch die Leistung eines reinen SCSI-Systems.

Von den Geschwindigkeitsangaben der Hersteller ließen wir uns nicht beeindrucken. Die Geschwindigkeiten in den Spezifikationen der Laufwerke können sowohl beim Lesen als auch beim Schreiben ohnehin nicht (ganz) erreicht werden. Man kann sich ihnen, wie wir feststellten, allerhöchstens dann am besten annähern, wenn man mit dem System nichts anderes tut, als CDs zu beschreiben - aber das verdirbt natürlich die Multitaskingeigenschaften, die man an OS/2 so schätzt. Folgerichtig interessierte uns auch, von welcher Güte die erzeugten CDs sind, wenn man das System während des Schreibens wie sonst belastet.

Alle Recorder wurden unter Warp 3 (mit FP 38) und Warp 4 (mit FP 9) gestetet. Als Software zum Beschreiben benutzten wir den *RSJ-CD-Writer* und *CDRecord/2*. Beide Programme weisen von Haus aus eine geringe Belastung des Systems während Schreibvorgängen auf und eignen sich daher ideal für OS/2.

Wir stellten einen recht heterogenen Datensatz aus unterschiedlich langen Dateien zusammen. Dazu gehörten kurze Text- und Bilddateien bis hin zu großen Bitmaps und auch Audiodaten, die mit einem Tagebuchprogramm erstellt worden waren. Diese großen Dateien waren etwa 6 bis 32 MByte groß. Insgesamt schrieben wir auf Rohlinge von *BASF* und *Maxell* mit allen Recordern jeweils 408 MByte Daten dieser Art unter den oben genannten Bedingungen und prüften anschließend die Qualität der beschriebenen *CD-Rs*. Danach begannen wir, noch zwei Sessions zusätzlich auf die CD zu brennen und kontrollierten die Brauchbarkeit der beschriebenen Rohlinge erneut. Als Test-CD-ROM-Laufwerke verwendeten wir gängige Modelle (z.B. das *Toshiba XR600B* ein ATAPI-CD-ROM Laufwerk), aber auch ältere wie das *Mitsumi FX001D*.

---

7 Für den Test des HP-Gerätes konfigurierten wir den Rechner entsprechend als reines IDE-System.

Ein paar Worte zur Geschwindigkeit

Die Schreibgeschwindigkeit der vorgestellten Laufwerke reicht von 2x bis 8x (beim *Plextor*). Diese Geschwindigkeiten werden aber in der Praxis nicht exakt erreicht. Das galt übrigens für alle getesteten Laufwerke. Jedoch liegt es nicht nur am Recorder, wie schnell das System aufzeichnet, sondern:

1. *An der CD-Writer Software*

   Es gibt zwei Arten, Daten auf eine CD zu schreiben. Bei der ersten benutzt man einen Schreibpuffer: Da zumindest eine Datei nicht auf mehrere Tracks verteilt werden darf, wird zunächst der Schreibpuffer der Software gefüllt, um ein kontinuierliches Schreiben zu ermöglichen. Erst dann wird der Inhalt auf die CD geschrieben. Währendessen wird der Schreibpuffer wieder gefüllt. Das bedeutet, daß sich die Schreibgeschwindigkeit eines Recorders nur dann besonders bemerkbar macht, wenn man einen Puffer füllen würde, der so groß ist, daß er den gesamten zukünftigen Inhalt der CD aufnehmen kann, um ihn dann in einem Zug auf die *CD* zu schreiben. Ansonsten unterbricht der Recorder den Aufzeichnungsvorgang, damit die Software den Schreibpuffer wieder füllen kann. Daher brauchten wir bei nahezu allen CD-Recordern mit dem *RSJ CD-Writer*, der nach diesem Prinzip arbeitet, etwa 25 bis 30 Minuten, um die Test-CDs zu beschreiben[8].

   Die zweite Möglichkeit besteht darin, ein sogenanntes Image der Daten zu erstellen, die auf die *CD-R* kopiert werden sollen. In einem solchen Falle kann man die Daten auf der *CD* in einem Track unterbringen und die Schreibgeschwindigkeit des Recorders besser ausnutzen. Wir zeichneten mit diesem Verfahren die vorbereiteten Daten mit den Geschwindigkeiten 2x in etwa 35 Minuten, 4x in ca. 25 Minuten und 8x in ca. 15 Minuten auf.

2. *An der Gesamtleisung des Systems*

   Sollte die Software mit Schreibpuffern arbeiten, ist der Aufzeichnungsvorgang um so schneller beendet, je schneller das System den Puffer zu füllen vermag. Ein langsames Festplattensystem wird den Schreibvorgang erheblich in die Länge ziehen. Die Leistung des Festspeichersubsystems ist v.a. bei der ersten Aufzeichnungsmethode relevant.

   Erzeugt man ein Image und brennt dieses auf die *CD-R*, ist es wichtig, daß der Datenstrom zum Recorder nicht abreißt. Um mit dem System wie gewohnt weiterarbeiten zu können, ist es erforderlich, daß der Rechner eine entsprechend hohe Gesamtleistung zeigt. Sind diese Rahmenbedingungen nicht gegeben, muß man das System für den Zweck des CD-Recordings dedizieren.

Wir benutzten beide Aufzeichnungsmethoden. Während der Vorstellung der CD-Writer-Software gehen wir noch detailliert auf die Aufzeichnungsverfahren ein.

Die Laufwerke im Detail

Im folgenden stellen wir Ihnen die oben aufgeführten Laufwerke vor. Die Vorstellung ist eingeteilt in eine Beschreibung

8 Diese Angaben verstehen sich inkl. der Zeit, die für den Abschluß der Session benötigt wird (stets etwa 5 bis 7 Minuten mit leichten Unterschieden von Gerät zu Gerät innerhalb dieser Zeitspanne).

der Recorder und deren Verarbeitung, die Erfahrungen während des Betriebes und einer Betrachtung der beschriebenen *CD-Rs*. Am Ende der Einzelvorstellungen finden Sie eine Zusammenfassung und eine Tabelle mit den einzelnen Leistungsmerkmalen und Preisen.

Teac CD-R55SK

Der *Teac CD-R55SK* ist ein Recorder mit schlichtem Design, den Sie auf alle SCSI-IDs konfigurieren können (standardmäßig ist der Recorder auf ID 3 eingestellt). Wie die Wahl der ID erfolgt auch die Terminierung über Jumper, die gut zugänglich sind, da sie vom Gehäuserand etwas überstehen, so daß man sie leicht ergreifen kann. Direkt daneben befindet sich die Audio-Out-Buchse zur Verbindung des Recorders mittels eines mitgelieferten Audiokabels mit der Soundkarte für das Abspielen von *Audio-CDs*. Auch die SCSI- und Stromversorgungsbuchsen sind gut zugänglich, da sie direkt mit der Gehäuserückseite abschließen. Die Belegung der einzelnen Jumper findet man auf einem kleinen Aufkleber auf der Gehäuserückseite.

Die vordere Abdeckung ist allerdings etwas wackelig, und der Schubladenverschluß klappt beim Öffnen heraus, bevor die Schublade ausfährt, was uns nicht gefiel, da es mehrere bewegliche Teile gibt, und damit die Fehleranfälligkeit des Gesamtsystems steigt (zumal die Schublade ja sehr häufig ein- und ausgefahren wird). Für diese am meisten beanspruchte Bauteilgruppe ist das Verschlußsystem zu diffizil. Unter der Schublade befindet sich wie üblich Kopfhöreranschluß und Lautstärkeregler. Auf der rechten Seite ist eine LED eingebaut, die sämtliche Aktivitäten des Recorders anzeigt, daneben der Knopf zum Öffnen und Schließen der Schublade. Angenehm ist hier, daß sich die Schublade auch durch einen leichten Stoß schließen läßt, was immer von Vorteil ist, da man bei geöffneter Schublade schlecht an den Eject-Knopf herankommt. Um das Laufwerk im Notfall öffnen zu können, befindet sich an der Vorderseite ein kleines Loch, in das man einen Draht einführen kann. Das ist recht umständlich, jedoch sollte der Fall, daß sich die *CD* nicht mehr aus dem Laufwerk entfernen läßt, kaum auftreten. Bis auf das Gerät von *Philips* verfügen übrigens alle Geräte über diesen Notmechanismus.

Im Betrieb zeigte sich das *Teac* zunächst nicht sehr überzeugend. Das Gerät erzeugt beim Schreiben dann und wann merkwürdig klingende Kratzgeräusche. Das muß nichts sagen, jedoch wird man stutzig. Zudem traten während des Schreibvorganges regelmäßig Fehler auf, und zwar mit beiden Testsystemen in allen beschriebenen Konfigurationen und mit beiden Recording-Programmen. Ließ man das System während der Aufzeichnung unberührt, vollrichtete der Recorder ohne Anstand seinen Dienst, jedoch widerspricht so eine Vorgehensweise dem Sinn eines Multitasking-Betriebssystems. Unser Ziel war ja herauszufinden, mit welchen Geräten man gefahrlos eine brauchbare CD im Hintergrund beschreiben kann - mit dem *Teac*-Recorder jedenfalls nicht; denn wir wiederholten den Aufzeichnungsvorgang mit neuen Rohlingen, auch anderer Hersteller, ohne daß der Recorder fehlerfrei aufzeichnete, wenn mit dem System noch gearbeitet wurde.

Trotz der Fehler, die beim Schreiben auftraten, konnten wir die beschriebenen *CD-Rs* in allen CD-ROM-Laufwerken lesen, in alten wie in neuen Modellen - solange wir es mit einer *CD* zu tun hatten, auf der sich alle Daten in einer Session befanden. Eine Multisession-

CD konnten wir nur bedingt lesen, zu allem Überdruß in den modernen CD-ROM-Laufwerken nicht. Der hartnäckige Versuch, doch von der *CD-R* zu lesen, endete in einem Trap. Wir vermuteten einen schwerwiegenden Fehler während der Aufzeichnung bzw. einen mangelhaften Rohling. Eine neu angefertigte *CD-R*, bei der auch während des Beschreibens keine Fehler auftraten, da wir das System für die Zeit der Aufzeichnung in Ruhe ließen, zeigte aber genau den gleichen Effekt. Wir stellten fest, daß nicht das ein oder andere Fixpak an diesem Fehler schuld sein konnte. Vielmehr war es das CD-ROM-Laufwerk, das die *CD* anscheinend nicht mochte (merkwürdigerweise auch eines von *Teac*). Wir stellten außerdem fest, daß dieser Fehler nicht auftrat, wenn das Laufwerk mit geringer Geschwindigkeit von der *CD* las. So konnten wir die Multisession-CD nur mit einem alten *2x-Mitsumi-CD-ROM-Laufwerk* lesen. Das ist aber ein denkbar schlechter Workaround, weil wohl kaum noch jemand ein solches CD-ROM-Laufwerk verwendet (wir machten es nur für unseren Test, um die Brauchbarkeit der *CD-Rs* auch mit älteren (aber multisessionsfähigen) Laufwerken zu testen). Da der Fehler immer mit Multisession-CDs auftrat, egal unter welchen Bedingungen aufgezeichnet wurde; das Laufwerk aber ansonsten korrekt arbeitete, und die Software mit allen anderen Laufwerken ebenfalls, kann der Fehler nur in einer Unverträglichkeit des Laufwerkes mit der Software oder auch nur am Laufwerk liegen. In beiden Fällen hatten wir keine andere Möglichkeit, als das *Teac* für Multisession-CDs nicht zu verwenden. Dieser Fehler muß mit anderen System nicht auftreten; bei uns trat er aber auf, so daß wir dem *Teac* nur sehr mißtrauisch einen neuen Rohling anvertrauten.

Philips CDD3600

Das *Philips CDD3600* war das kostengünstigste Gerät im Test, aber das sollte kein Vorurteil sein. Der Recorder ist lediglich auf die SCSI-IDs 1,2 und 4 konfigurierbar (Standard ist hier 4) und -wie üblich- terminierbar durch Jumper, die etwas weniger gut erreichbar sind als beim *Teac*-Laufwerk, jedoch gerade noch mit den Fingernägeln von den Steckern entfernt und umgesetzt werden können. Anweisungen zum Setzen der **Jumper** befinden sich in der Gehäuserückseite eingestanzt. Direkt daneben findet man die Stromversorgung, die SCSI-Buchse und den Audio-Ausgang. Bis auf die Stromversorgung schließen ebenfalls alle Buchsen mit der Gehäuserückseite ab, so daß sie leicht erreichbar sind. Bei der Stromversorgung ist der Stecker allerdings ohnehin etwas länger, so daß ein Einführen in die Buchse problemlos möglich ist.

Bei Betrachtung der Verarbeitung fiel uns auf, daß die Schublade ein wenig wackelig ist; sie läßt sich auch im ausgeschalteten Zustand des Laufwerkes leicht öffnen (was der Hersteller allerdings beabsichtigte). Dazu greift man in eine kleine Bucht unterhalb der Schublade und kann sie relativ einfach aus dem Laufwerk ziehen. Unterhalb der Schublade befindet sich auf der rechten Seite ein kleiner roter Knopf zu deren Bedienung, der ein wenig aus der Abdeckung hervorschaut. Er kann besser als beim *Teac* und auch den anderen Laufwerken (bis auf das HP-Gerät) ertastet und benutzt werden. Auch kann man die Schublade durch einen kleinen Schubs dazu veranlassen, wieder in das Laufwerk einzufahren. Das Ein- und Ausfahren läuft dabei sehr ruhig und beinahe geräuschlos ab.

Das Modell von *Philips* verfügt an der Vorderseite über zwei LEDs, wovon eine signalisiert, ob sich eine *CD* im

Laufwerk befindet (grün). Lesevorgänge werden durch ein Blinken dieser LED angezeigt. Eine zweite (rot) befindet sich in der Mitte der Vorderseite und leuchtet nur, wenn das Gerät aufzeichnet. Auf der linken Gehäusevorderseite findet man wie bei allen anderen Laufwerken den Lautstärkeregler und die Buchse für den Mikrophonausgang. Die Solidität der Verarbeitung ist im großen und ganzen der des *Teac*-Recorders gleichsetzbar; das Gerät macht jedoch in der Ausführung der Bedienelemente einen besseren Eindruck.

Die Arbeit mit dem *Philips* verlief angenehm und ohne Probleme. Selbst bei starker Systemauslastung zeichnete das Gerät ohne Fehlermeldungen auf. Die angegebene, recht niedrige 2-fache Schreibgeschwindigkeit fiel jedoch kaum ins Gewicht, allerhöchstens wenn wir mit Images arbeiteten, um unsere *CDs* herzustellen.

Das Gerät arbeitete vollkommen ruhig ohne irritierende Geräusche und erwärmte sich selbst bei längerem Betrieb kaum. Allgemein kann man dem *Philips* ein ruhiges Laufverhalten zusprechen. Auch mit der Schublade gab es keinerlei Probleme, wenngleich sie zunächst etwas wackelig erschien und sich auch per Hand wie beschrieben ausfahren läßt (ist der Recorder erst einmal eingebaut, hat man ein gut verarbeitetes Gerät im System).

Die mit dem Recorder hergestellten *CD-Rs* weisen eine vollkommen gleichmäßige Struktur der beschriebenen Oberfläche auf. Ringstrukturen oder sonstige Spuren gab es nicht. Daneben ließen sich die *CD-Rs* in allen CD-ROM-Laufwerken problemlos lesen. Auch die hergestellten Multi-Session-CDs bereiteten keine Schwierigkeiten, wie wir sie beim Modell von *Teac* beobachtet hatten.

Im Gegensatz zu dem vorherigen und dem noch folgenden Modell in unserem Test kann das Gerät von *Philips* auch *CD-RWs* bearbeiten. Die hergestellten *CD-RWs* konnten wir ebenso von den zur Verfügung stehenden CD-ROM-Laufwerken lesen, nur von den älteren Modellen nicht. Das Löschen und Wiederbeschreiben verlief ohne nennenswerte Besonderheiten unter allen genannten Systemkonfigurationen (im übrigen auch mit den anderen Recordern, die *CD-RWs* verarbeiten können), so daß wir nicht näher darauf eingehen müssen.

Sony CDU 948 S

Das *Sony* ist das einzige Gerät, das mit der Caddy-Technologie arbeitet. Ein Caddy liegt dem Laufwerk glücklicherweise bei. Der Caddy selbst ist sehr leicht und recht empfindlich; die Handhabung ein wenig gewöhnungsbedürftig (da man in der Regel das Arbeiten mit Schubladen gewöhnt ist, die sich auf dem Markt auch durchgesetzt haben). Unter der Laufwerksöffnung befindet sich ein Knopf zum Ausgeben des Caddys und eine LED, die sämtliche Laufwerksaktivitäten anzeigt. Sie leuchtet grün, sobald das Gerät mit Strom versorgt wird, wechselt die Farbe aber während eines Aufzeichnungsvorganges von Grün nach Orange. Das ist nicht so komfortabel wie beim *Philips*, aber doch recht angenehm, weil man immer weiß, was das Gerät gerade tut. Eine Eigenschaft, die wir übrigens beim *Teac*-Recorder vermißten, bei dem die LED immer blinkte, egal ob das Gerät gerade am Schreiben oder am Lesen war.

Wie an den anderen Laufwerken auch befindet sich an der linken Gehäusevorderseite ein Audio-Ausgang und ein Lautstärkeregler. An der Oberseite der Gehäuserückseite findet man den

Audioausgang und die **Jumper** zur Terminierung und Setzen der SCSI-ID, die standardmäßig auf ID 6 gejumpert ist. Das wird man in den meisten Fällen ändern müssen, da die IDs 6 und 5 in aller Regel bereits von den Festplatten belegt sind.

Die Platine, auf der sich die Anschlüsse befinden, ist etwas wackelig im Laufwerk montiert, jedoch ist dies bei den anderen bereits getesteten Laufwerken auch so (lediglich das *Yamaha*-, *Plextor*- und *HP-Gerät* machen in dieser Hinsicht eine Ausnahme). Über der SCSI-Buchse befinden sich zwei Endwiderstände, die auf die Platine gesteckt sind. Vom Standpunkt der Austauschbarkeit ist dies natürlich von Vorteil; aus Sicht der Verarbeitung ist so etwas allerdings mangelhaft, da die kleinen Füßchen der Widerstände recht dünn sind und leicht abbrechen können. Beim Umgang ist also Vorsicht angesagt. Die Gehäuseoberseite ist durchlöchert, so daß für eine gute Durchlüftung während des Betriebes gesorgt ist.

Der Caddy läßt sich einfach in das Laufwerk einführen. Die mit *CD Caddy* beschriftete Verschlußklappe wird dabei gegen eine mit *Caddy loaded* ausgetauscht. Das bringt jedoch nicht viel, es sei denn der Rechner steht als Desktopgehäuse auf dem Schreibtisch. Ansonsten kann man die Beschriftung, die nur in den Kunststoff der Verschlußklappe eingestanzt ist, nicht lesen (wäre die Schrift schwarz oder eine Disk-In LED vorhanden, wäre das von Vorteil, es sei denn man kriecht gerne unter dem Schreibtisch herum).

Auffällig war, daß das Laufwerk bereits recht warm wurde, wenn man nicht damit arbeitete. Die Wärmeentwicklung steigerte sich während des Betriebes auch noch recht stark, womit die Löcher zur Durchlüftung an der Gehäuseoberseite verständlich werden. An sich ist das nicht sonderlich nachteilig, jedoch wird auch der CD-Rohling überdurchschnittlich warm. Theoretisch kann das unvorteilhaft sein, hatte allerdings keine Auswirkungen auf die Qualität der beschriebenen *CD-Rs*.

Ab und an gibt das Gerät einige Geräusche beim Schreiben von sich, was wie beim *Teac* auf eine nicht so fein verarbeitete Mechanik schließen läßt, die den Laser zum Schreiben und Lesen im Laufwerk bewegt.

Die beschriebenen *CDs* wiesen dieselbe, gleichmäßige Oberflächenstruktur auf, wie die Rohlinge, die mit dem *Philips*-Recorder verarbeitet wurden und ließen sich -auch in der Multi-Session Variante- von allen CD-ROM-Laufwerken lesen. Da der *Sony*-Recorder wie das Modell von *Teac* ebenfalls mit vierfacher Schreibgeschwindigkeit aufzeichnet, konnte der Schreibvorgang bei der Verwendung von Images gegenüber dem Modell von *Philips* abgekürzt werden. Fehler traten während des Schreibens auch bei starker Systemauslastung nicht auf. Das *Sony*-Modell macht daher einen recht guten Eindruck, sofern man einige Mängel an der Verarbeitung außer acht läßt und sich mit Caddies anfreunden kann.

Plextor PX-R820Ti

Das Plextor ist von allen bereits betrachteten Recordern am stabilsten verarbeitet. Die einzelnen Buchsen für den Audioausgang, den SCSI-Bus und die Stromversorgung sind fest in das Gehäuse eingearbeitet und wackeln nicht, was natürlich gut ist, da das Gerät dadurch sehr stabil und widerstandsfähig wird. Die Terminierung und das Setzen der SCSI-ID wird wie bei allen anderen Laufwerken über Jumper vorgenommen (beim *PX-R820Te*, der externen Variante übrigens mit mit Hilfe

eines Drehschalters). Als SCSI-IDs kommen 7 bis 0 in Frage. Da 7 stets für den Hostadapter belegt ist, kann man das Gerät optimal an die Verhältnisse seines SCSI-Bus anpassen, was bis auf den *Philips* allerdings alle anderen Geräte auch erlauben. Standardmäßig ist der *Plextor-Recorder* auf die ID 4 konfiguriert. Um einige bestimmte IDs zu setzen, braucht man zum Teil mehr als einen Jumper, der bereits in der Steckerleiste sitzt. Zusätzliche Jumper sind jedoch im Lieferumfang des Laufwerkes enthalten.

Die Schublade ist robust und wackelt kaum. Neben den Standardelementen Kopfhörerausgang und Lautstärkeregler befinden sich gleich vier LEDs an der Stirnseite des Recorders, zwei für Schreib- und zwei für Lesevorgänge: Die *ON/DISC* Anzeige signalisiert, daß das Gerät mit Strom versorgt wird; *DISC* blinkt, falls von einer *CD* gelesen wird. Die *L/H*-Anzeige zeigt an, mit welcher Geschwindigkeit der Recorder gerade aufzeichnet. Die orangene LED *L* leuchtet wenn mit 1-,2- oder 4-facher Geschwindigkeit, *H* wenn mit 8-facher Geschwindigkeit geschrieben wird. Daneben gibt es zwei Knöpfe zur Steuerung: Einer zum Ein- und Ausfahren der Schublade; und einen anderen zum Abspielen von Audio-CDs und dem Wechseln der einzelnen Tracks. Inwiefern man diesen Knopf braucht, sei einmal dahingestellt, da man *CDs* wohl eher mit der Audio-CD-Spieler-Software abspielen wird, als ständig zum Laufwerk zu greifen. Das Ganze kann vielleicht praktisch sein, wenn man über keine solche Software verfügt, und über den Recorder auf die *Audio-CD* zugreifen möchte. Die Frage ist nur, wer das macht, und ein CD-Recorder braucht dieses Feature sicherlich nicht.

Daneben ist der Recorder leider alles andere als ein Leisetreter: Bereits die Schublade fährt unter deutlicher Geräuschentwicklung ein und aus. Das ist allerdings nicht so tragisch. Ein wirklicher Störenfried ist jedoch ein kleiner Ventilator an der Rückseite des Gehäuses, der natürlich auf der einen Seite dafür sorgt, daß sich das Gerät kaum erwärmt; auf der anderen Seite gibt er dabei allerdings ein hochfrequentes Summen von sich, das leider auch nicht im allgemeinen Rauschen des Netzteillüfters des Rechners untergeht. Das sorgt für ein nicht sehr angenehmes Arbeitsklima.

Das Gerät zeichnet im Betrieb mit dem *RSJ-CD-Writer* in aller Regel mit vierfacher Geschwindigkeit auf, nur ab und an mit dem Maximum (8-fach). Es hängt hier im entscheidenden Maße davon ab, ob auch der Datenstrom, der für eine solche Schreibgeschwindigkeit nötig ist, vom System zur Verfügung gestellt werden kann. Bei entsprechendem Systemausbau und bei der Verwendung von Images beschrieb der *Plextor* tatsächlich mit 8-facher Geschwindigkeit, was sich natürlich entsprechend bemerkbar machte. Sobald man jedoch wieder mit dem System ganz normal arbeitet, wird mit 4-facher Geschwindigkeit weitergeschrieben. Wenn man also nicht gerade ein System betreibt, das neben einer guten Ausstattung fast ausschließlich für das CD-Recording gebraucht wird, bringt einem die hoche Aufzeichnugnsgeschwindigkeit, mit dem sich das Modell von Plextor über alle anderen erhebt, gar nichts; es sei denn man arbeitet mit Images zum Brennen, nicht mit Schreibpuffern.

Die erzeugten *CD-Rs* sind wie bei den beiden Laufwerken zuvor von nicht zu beanstandender Qualität. Wir konnten Sie in allen CD-ROM-Laufwerken lesen, auch wenn sich mehr als eine Session auf der *CD* befand, bzw. die CDs mit der Maximalgeschwindigkeit beschrieben

worden sind (bzw. teilweise mit dieser Geschwindigkeit).

Im großen und ganzen beindruckte das Edelstück von *Plextor* dennoch nicht so recht. Es enthält einige Schnörkel, die man niemals braucht aber mitbezahlt. Daneben läßt sich die hohe Schreibgeschwindigkeit nur mit schnellen Systemen und einem nahezu nur auf das CD-Recording beschränkten Arbeitsverhalten ausnutzen, sofern man *CDs* »on the fly« herstellt und nicht mit einem Image. Auch die wirklich gute Verarbeitung trägt nicht zu einem überaus befriedigenden Urteil bei.

Yamaha CRW-4416S

Das *Yamaha* ist ein schlichtes Modell, das vom Äußeren her nicht den Anschein macht, es könnte sich über die anderen erheben. Aber nur dem Äußeren sollte man eben nicht trauen. Die Verarbeitung ist nämlich gut, wenngleich hier und da einige Abstriche gegenüber dem *Plextor*- und *HP*-Modell gemacht werden müssen, die aber nicht ins Gewicht fallen. Wie das *Plextor* besitzt es einen kleinen Ventilator zur Kühlung des Laufwerkes während des Betriebs. Im Gegensatz zum teuersten Gerät im Test stört dieser Ventilator aber nicht durch ein hochfrequentes Summen (der Beweis dafür, daß so ein Winzigventilator nicht nervtötend sein muß). Das *Yamaha* läßt sich über Jumper ebenfalls auf jede SCSI-ID einstellen (Standard ist hier 3). Geliefert wird das Gerät in der Kit-Version mit einem SCSI-Flachbandkabel, einem Audiokabel, Handbuch, Schrauben zum Einbau, einem Ersatzjumper sowie einer *CD-R* und einer *CD-RW*. Betrachtet man den recht hohen Preis einer *CD-RW* ist das eine nette Dreingabe, da man gleich nach dem Auspacken des Gerätes und der Installation der Software mit dem Schreiben beginnen kann.

Auch das *Yamaha-Modell* zeigt durch seine LEDs dem Benutzer, was es gerade tut. Durch einen Farbwechsel von Grün nach Rot wird man über zur Zeit laufende Schreibvorgänge informiert. Das Aufzeichnen vollzieht sich wie beim *Philips-Recorder* völlig geräuschlos, wobei sich das Gerät nur mäßig erwärmt. Die etwas höhere Schreibgeschwindigkeit (4-fach) macht sich wieder nur bei dem besser ausgebauten, reinen SCSI-System bemerkbar. Auf jeden Fall traten während des Beschreibens der *CD-R* mit diesem Laufwerk keinerlei Fehler auf, und die *CDs* waren von allen Laufwerken zu lesen.

Das gilt auch für die *CD-RWs*, die sich wie beim *Philips*- und *HP*-Modell ohne Schwierigkeiten verarbeiten ließen und bis auf die alten Laufwerke von allen zur Verfügung stehenden CD-ROM-Laufwerken gelesen werden konnten. Auch für diesen Bereich ist das *Yamaha* geeignet, allerdings hatten auch die beiden anderen *CD-RW*-fähigen Recorder keine Schwierigkeiten mit den wiederbeschreibbaren Medien. Herausstechend ist nur, daß das Modell von *Yamaha* *CD-RWs* ebenfalls mit 4-facher Geschwindigkeit beschreibt, so wie normale *CD-Rs*, was die anderen beiden Geräte nicht können. Bei entsprechender Arbeitstechnik macht sich dieses Merkmal als kleiner Vorteil gegenüber dem *Philips* und *HP* bemerkbar. Alles in allem ist das *Yamaha* also ein durchaus empfehlenswertes Produkt, das man relativ zur gebotenen Qualität und Leistungsvielfalt zu einem akzeptablen Preis bekommt.

HP SurerStore 8100i

Das *HP*-Gerät besticht durch eine gewisse Eleganz im Design und eine gute Verarbeitung. In dieser Hinsicht steht es mit dem *Plextor* auf einer Ebene, besticht aber durch weitaus mehr

Funktionalität. Das Gerät gibt es übrigens auch für den SCSI-Bus, so daß man auch als SCSI-Fan in den Genuß dieses Laufwerkes kommt. Die stabile Verarbeitung zeigt sich auch im Umgang mit dem Gerät, so daß man keine Mangelware erhält. Das Laufwerk wird in der Kit-Variante mit einem IDE-Kabel, Schrauben zum Einbau, einem dicken Handbuch, Software sowie, genau wie beim *Yamaha*, einer *CD-R* und einer *CD-RW* geliefert. Mit der Software kann man unter OS/2 nichts beginnen, aber das stört nicht sehr, weil der OS/2-Anwender für das CD-Recording ohnehin seine native Software hat.

Das Gerät läßt sich über sehr gut zugängliche Jumper auf Master oder Slave einstellen (Slave ist Standard). Wir konnten den Recorder allerdings nicht in jeder Konfiguration in unserem schnelleren Testsystem betreiben, jedoch immer als Master am 2. IDE-Channel. Am ersten Channel als Slave konfiguriert wollte unser schnelles System das Laufwerk allerdings nicht so recht akzeptieren. Das zusätzlich angeschlossene IDE-CD-ROM-Laufwerk wurde mit in diesem Falle ebenfalls unbrauchbar, warum konnten wir nicht herausfinden. Dieser Effekt muß im Controller begründet gewesen sein. Der alte Rechner akzeptierte das Laufwerk nämlich in allen Konfigurationen. Falls Sie ebenfalls auf ein derartiges Problem stoßen sollten, vermuten Sie also nicht sofort einen Fehler im Laufwerk, sondern versuchen Sie einfach eine andere Konfiguration an den IDE-Kanälen.

Mit dem Recorder zu arbeiten, machte Freude. Er erwärmt sich und den gerade zu verarbeitenden Rohling nur mäßig, ist nahezu geräuschlos und zeigt selbst am IDE-Port überraschend gute Leistungen. Zusammen mit der soliden Verarbeitung wird der *HP*-Recorder für den anspruchsvollen Anwender sehr interessant.

Zudem zeichnet er ohne Fehler auf, egal wie stark das System gerade ausgelastet ist. Leider lag uns zum Test nur die IDE-Variante vor, so daß wir mit diesem Gerät nicht mehr in den Genuß des »SCSI-Effektes« kommen konnten (daß man den Aufnahmevorgang nämlich gar nicht merkt, weil er tatsächlich im Hintergrund abzulaufen scheint). Jedoch war die Belastung der CPU auch beim IDE-Gerät nicht so hoch, daß man mit dem System nichts mehr hätte anfangen können. Die Schwächen von IDE traten zu Tage, jedoch war das Antwortverhalten gerade noch akzeptabel, so daß auch die IDE-Variante des *HPs* eine Empfehlung verdient.

Die mit dem *HP* beschriebenen CDs verursachten keine Probleme beim Lesen durch unsere CD-ROM-Laufwerke, auch die mit dem Laufwerk verarbeiteten *CD-RWs* nicht (bis auf die Einschränkung, daß die älteren CD-ROM-Laufwerke die *CD-RWs* nicht lesen konnten). Das *HP* ist daher ein empfehlenswertes Gerät, das sich für den Einsatz unter OS/2 sehr gut eignet und bildet die Spitze der Laufwerke, die wir Ihnen als Sieger dieses Tests im folgenden vorstellen werden.

Das Resümee

Ein Gerät hat uns gar nicht gefallen, nämlich das Modell von *Teac*. Die Verarbeitung ist zwar akzeptabel, aber beim Schreiben treten regelmäßig Fehler auf, und zwar auf beiden Testsystemen, besonders dann, wenn man das System nicht ausschließlich zum Beschreiben der *CD-R* dediziert, sondern nebenbei noch weiterarbeiten möchte. Ferner sind die erzeugten *CDs* im großen und ganzen von keiner sonderlich großen Güte. Daß das Gerät unangenehme Kratzgeräusche beim Schreiben verursacht,

mag nur eine Nebensächlichkeit sein, trägt aber zum ohnehin schon angeschlagenen Ansehen nicht zu einer Empfehlung bei. Letztlich kann man auch bessere Geräte dieser Preisklasse finden, so daß wir dem *Teac* keine Kaufempfehlung geben können.

Das *Sony* ist ein akzeptables Laufwerk, weist jedoch den Nachteil auf, daß es mit der Caddy-Technologie arbeitet. Die gefertigten *CDs* haben zwar eine hohe Güte, allerdings ist das Hantieren mit dem Caddy etwas umständlich. Das Gerät wird während des Betriebes recht warm, für eine gute Durchlüftung ist allerdings gesorgt. Ansonsten spricht nichts gegen das Sony-Laufwerk - vorausgesetzt man hat nichts gegen Caddies.

Das *Plextor* ist das teuerste Gerät in unserem Test gewesen. Was es gegenüber den anderen Geräten (ausgenommen dem von *HP*) besonders unterscheidet, ist die weitaus solide Verarbeitung, die man für fast neunhundert Mark allerdings auch verlangen kann. Die Möglichkeit, *CDs* mit achtfacher Geschwindigkeit zu beschreiben, ist natürlich reizvoll; jedoch sollte man dabei beachten, daß *CD-Rs*, die man in diesem Tempo beschreibt, nicht die Güte haben können wie solche, für die man sich etwas mehr Zeit läßt. Wer viele *CDs* in kurzer Zeit beschreiben möchte, der sollte sich das *Plextor* etwas näher ansehen. Beachten Sie aber in jedem Fall, daß Sie bestimmte *CDs* mit festgelegter Geschwindigkeit beschreiben müssen (z.B. wenn Sie Kopien von *Audio-CDs* anfertigen); ferner, daß die Zeit, die Sie für die Herstellung einer *CD* brauchen, vielmehr vom Ausbau Ihres Systems abhängt als vom Recorder. So kann Ihnen bei geringem Hauptspeicherausbau und ungünstigerer Konfiguration ein Gerät, das mit 8-facher Geschwindigkeit aufzuzeichnen vermag, gar nichts nützen. Was das *Plextor* ein wenig unangenehm macht, ist das hochfrequente Summen des kleinen Lüfters. Mit der Zeit geht einem das ständige »Gesumse« auf den Geist, so daß man es am liebsten abschalten möchte - was leider nicht möglich ist. Für den Endanwender ist das *Plextor*-Gerät außerdem zu teuer, v.a. dann, wenn man nur ab und an eine *CD* beschreibt. Die fehlende Möglichkeit, auch *CD-RWs* verarbeiten zu können, stört außerdem, wenn man den hohen Preis beachtet.

Die Geräte, die uns am besten gefallen haben, waren die Laufwerke von *Yamaha*, *HP* und *Philips*. Alle drei erlauben das Beschreiben von *CD-Rs* **und** *CD-RWs* und zeigen allesamt gute Leistungen. Die Laufwerke eignen sich auch für den Einsatz in kleinen Netzwerken zu Backup-Zwecken und dürften den OS/2-Privatanwender am ehesten befriedigen - auch hinsichtlich des Preises.

Das *Philips* ist für den Endanwender mit schmalem Geldbeutel oder wenn man nicht allzuviel in einen CD-Recorder investieren möchte, das optimalste Gerät, verkaufen kann man sich hier nicht. Es kann zwar laut Angabe nur mit Doublespeed Schreiben (*CD-Rs* und *CD-RWs*), dafür beschreibt es aber beide CD-Typen und zwar sehr zuverlässig. Da die Aufzeichnungsgeschwindigkeit des Recorders wirklich nur bei dedizierten oder nahezu dedizierten System zum Vorschein tritt, läßt sich innerhalb von etwa 35 bis 40 Minuten eine komplette *CD* mit dem Laufwerk beschreiben. Fehler traten beim Schreiben nicht auf, selbst wenn wir das System stark beanspruchten. Das Schreiben selbst macht sich dank SCSI kaum bemerkbar, und selbst wenn man mit einem Hybridsystem (IDE/SCSI) arbeitet, verlieren die erstellten *CDs* nicht an Qualität (nur das Beschreiben dauert etwas länger,

was den Nachteil der IDE-Festplatte zum Vorschein bringt). In der Verarbeitung steht das *Philips* dem *Plextor*- oder dem *HP*-Gerät etwas nach, auch die geringe Lesegeschwindigkeit (6x) ist etwas enttäuschend, aber das fällt kaum ins Gewicht, vor allem wenn man den günstigen Preis bedenkt. Dieses Gerät ist daher eine optimale Kombination hoher Funktionalität, einfacher Anwendung und weist nur geringe Anschaffungskosten auf. Wer kein SCSI hat: Das *Philips CDD3600* gibt es auch in einer ATAPI-Variante (Typ *CDD3610*).

Das *Yamaha*-Gerät ist schon ein wenig teurer. Dafür erlaubt es höhere Aufzeichungsgeschwindigkeiten. Vor allem die Möglichkeit mit vierfacher Geschwindigkeit *CD-RWs* zu beschreiben, sticht hier hervor und hebt es vom *Philips* ab. Das Laufwerk erstellt *CDs*, die problemlos von allen uns zur Verfügung stehenden CD-ROM-Laufwerken gelesen werden konnten. Fehler beim Schreiben traten auch hier nicht auf. In der Verarbeitung ist es ein wenig besser als das *Phillips*, womit es all jenen empfohlen sei, die für ein bißchen mehr Leistung knapp dreihundert Mark mehr ausgeben möchten.

Mit dem *Yamaha* auf einer Stufe steht das *HP SureStore 8100i*, jedoch mit einer leicht besseren Verarbeitung. Das Laufwerk gibt es sowohl als ATAPI-Gerät, wie es uns zur Verfügung stand, als auch als SCSI-Laufwerk, was natürlich immer vorzuziehen ist. Der Recorder ist sehr solide verarbeitet und zeigt gute Leistungen. Die erzeugten CDs sind nicht zu beanstanden. Die 24-fache Lesegeschwindigkeit ist ein erhöhter Anreiz zum Kauf, wenngleich der Recorder nur mit Doublespeed *CD-RWs* verarbeiten kann. Dennoch erwirbt man mit dem *HP Sure Store 8100i* ein solides und qualitativ hochwertiges Gerät, das es mit dem *Plextor* in jeder Hinsicht spielend aufnehmen kann, v.a. wenn man den bedeutend geringeren Preis beachtet. Daneben denken wir nicht, daß eine hohe Aufzeichnungsgeschwindigkeit wirklich das Wichtigste ist.

Wir empfehlen Ihnen also die erwähnten drei Laufwerke. Welches für Sie in Betracht kommt, wird in erster Linie vom Einsatzzweck und dem Betrag abhängen, den Sie investieren wollen. Die wichtigsten Daten aller Recorder haben wir für Sie in der Tabelle *OS/2-taugliche CD-Recorder* zusammengefaßt. ❑

Typ	***Geschwin.***	***Bustyp***	***CD-RW fähig***	***Getestet***	***Wertung***	***Preis in DM (inkl. MwSt.)***
Teac CD-R 55S	4x, 12x	SCSI	nein	x	●	479.- (Bulk)
Philips CDD3600	*2x, 2x, 6x*	*SCSI*	*ja*	x	●●●	*379.- (Bulk)*
Philips CDD3610	2x, 2x, 6x	ATAPI	ja			429.- (Bulk)
Sony CDU 948 S	4x, 8x	SCSI	nein	x	●●	459.- (Bulk)
Plextor PX-R820Ti	8x, 20x	SCSI	nein	x	●●	899.- (Bulk)
Yamaha CRW-4416S	*4x, 4x, 16x*	*SCSI*	*ja*	x	●●●●	*649.- (Bulk)* *669.- (Kit)*
Yamaha CRW-4416E	4x, 4x, 16x	ATAPI	ja			649.- (Bulk) 669.- (Kit)
HP SuperStore 8100i	*4x, 2x, 24x*	*ATAPI*	*ja*	x	●●●●	*669.- (Kit)*

OS/2-taugliche CD-Recorder: Die Geschwindigkeiten sind angegeben in *Schreiben*, *Schreiben auf eine CD-RW* und *Lesen*. Vier Punkte in der Wertung spiegeln das beste, 1 Punkt das schlechteste Ergebnis wieder. Kursive Einträge stehen für unsere Empfehlung. Der Mehrpreis auf die Geräte für ein externes Gehäuse beträgt DM 139,00. Alle Preise sind Angebote der Fa. AP-COM.

AP-COM

Andreas Pieter Computer

Hardware, Software, Internet-Service

Telefon: 0 68 94/88 88 03

http://www.ap-com.de

Telefax: 0 68 94/88 88 04

RSJ CD-Writer für OS/2, Version 2.73
Vollversion (inkl. Handbuch und Disketten) DM 499,-
Internet Only (nur Freischaltungs-Code) DM 435,-
Update von Version 2.x DM 149,-
Update von Version 1.x DM 229,-

CD-(Re)Writer:
Philips CDD3600, 2x/2x/6x, SCSI, intern DM 379,-
Philips CDD3610, 2x/2x/6x, ATAPI, intern DM 429,-
Yamaha CRW-4416S, 4x/4x/16x, SCSI, intern DM 649,-
Yamaha CRW-4416E, 4x/4x/16x, ATAPI, intern DM 649,-
Plextor PX-R820Ti, 8x/20x, SCSI, intern DM 899,-
Mehrpreis für externes Gehäuse inklusive Einbau: DM 139,-
Mehrpreis für RSJ CD-Writer Internet-Only DM 420,-

Nur DM 15,- Versandpauschale

inklusive Nachnahme-Gebühr

AP-COM - Friedhofstraße 77 - 66386 St. Ingbert

CD-Recorder-Software für OS/2

Was native Software für Warp betrifft, so gibt es zwar für viele Bereiche eine Lösung, jedoch ist sie selten optimal: Um einen Browser zu benutzen, der HTML 3.2 versteht, muß man wohl oder übel zum Programmonster *Netscape* greifen; Standardanwendungen erhält man mit dem Programmpaket *StarOffice*. Beide Applikationen sind keine wirklich gelungenen OS/2-Anwendungen. Besonders bei Software aus dem Hause *StarDivision* merkt man, daß dort mit einer Objektbibliothek gearbeitet wird, die auch für die Windows Applikationen Verwendung findet, und so verhalten sich diese Anwendungen auch unter OS/2. Aber man hat eben leider selten die Wahl, so daß man -zumindest noch- zu diesen Applikationen greifen muß. Im Bereich des CD-Recordings ist es ähnlich, aber hier stehen mit der *CD-Writer Software* aus dem Hause *RSJ* und dem Freeware-Programm *CDRecord/2* Produkte für OS/2 zur Verfügung, die sich optimal für den Einsatz unter Warp eignen. Wir stellen Ihnen beide Anwendungen im folgenden Kapitel im Zusammenhang mit den getesteten Recordern vor und geben Tips und Tricks rund um die Programme.

RSJ-CD Writer

Unsere Entwicklungsphilosophie zielt auf schlanke Anwendungen ab. Um so erfreuter waren wir daher, als wir unsere Version des *RSJ CD Writer* zum Test bekamen: Das gesamte Programm befindet sich nämlich auf einer 1,44 MByte Diskette. Nach der Installation belegt es gerade einmal 2,2 MByte auf der Festplatte - so muß es sein. Das Programm ist also, schon bevor man sich näher damit beschäftigt hat, ein Beweis dafür, daß man heutzutage nicht die Festplatte mit riesigen Programmpaketen aufzufüllen braucht, sondern auch mit Verstand entwickeln kann. Im weiteren Test sollte uns der *RSJ CD-Writer* aber auch noch über diese angenehme Tatsache hinaus gefallen, wenngleich es hier und da einige Mängel gibt, die man sich in künftigen Versionen beseitigt wünscht. Bestimmte Probleme lassen sich jedoch bereits im Vorfeld durch einige Einstellungen vermeiden, so daß es sich mit dem Programm gut arbeiten lassen sollte. Einige Schwächen sind nur mit dem im Lieferumfang enthaltenen Programm *CDView* ausgleichbar.

Lieferumfang

Der *RSJ CD-Writer* besteht aus zwei Produkten: Einem speziellen *Dateisystem*, mit dessen Hilfe man auf eine *CD-R/RW* wie auf eine Festplatte schreiben kann; und ein Programm namens *CDView*, mit dem Operationen mit *CD-R/RWs* und normalen *CDs* auf Trackbasis möglich sind. Daneben ist dem Programmpaket eine Programmierschnittstelle beigefügt, die es erlaubt, eigene Applikationen zu schreiben, die ein *CD-ROM-Recording* auf der Basis des *RSJ-Dateisystems* ermöglichen. Man findet außerdem ein Handbuch zum Programm, das jedoch dringend einer Bearbeitung bedürfte, da es kein Musterbild an guter Gliederung ist. Die Dokumentation zum API muß man sich über die *RSJ-Homepage* via Download besorgen. Sie liegt dort als PDF-Datei auf den Downloadseiten des Softwarehauses. Die Telefonnummer, eMail- und Webadresse von *RSJ* finden Sie im Kasten *Kontakt mit RSJ*.

Die Installation

Wir testeten die Version 2.71 und 2.74 des *RSJ CD-Writers*. Die Installation (wie das Update) des Programmpake-

tes gestaltete sich durch das Installationsprogramm des Produktes einfach und dürfte keine Probleme verursachen, so daß wir uns auf einige Besonderheiten beschränken können, die man während der Installation beachten sollte. Verfahren Sie zur Installation wie folgt:

❶Starten Sie das Programm *install* auf der Installationsdiskette. Daraufhin erscheint ein Fenster wie in *Abb.1*.

❷Ändern Sie die Cachegröße für das Dateisystem. Wir machten mit Caches, die eine Größe von 80 bis 100 MByte hatten, die besten Erfahrungen.

❸Der Cache-Pfad weist auf den Pfad, der in Ihrer CONFIG.SYS mit der Umgebungsvariablen TMP angegeben ist. Hier werden die Pufferdaten abgelegt. Achten Sie darauf, daß sich dieses Verzeichnis auf einer Partition befindet, auf der noch genügend freier Speicherplatz vorhanden ist. Wir empfehlen Ihnen, den Cache-Pfad auf dem Standardwert zu belassen. Sollten Sie Ihn ändern wollen, achten Sie darauf, daß Sie den neuen Pfad dem Programm CDWFSD.EXE, das mit der Anweisung RUN in der CONFIG.SYS eingetragen ist, als Parameter übergeben, da der Pfad, wie wir feststellten, bereits beim Start von CDWFSD.EXE festgelegt sein muß. Die Anweisung sieht dann so aus:

```
RUN=<LW>:\CDWFS\CDWFSD.EXE
        -p "<LW2>:\NeuerCachePfad" [...]
```

Dabei ist LW das Installationslaufwerk des *CD-Writers* und LW2 das Laufwerk, auf dem sich das Verzeichnis für die Cache-Dateien befindet.

! ***Geben Sie dem Dateisystem einen Cache, der groß genug ist. Sollte auf dem Laufwerk, auf dem der Zwischenspeicher eingerich-***

RSJ CD-Writer Installation

RSJ SOFTWARE
CD-Writer Version 2.71

Dateisystem
Puffergröße 2048 KB Modus CD-XA
Cachegröße 20000 KB ISO-Level 3
Cache-Pfad h:\system

CD-Recorder
Geschwindigkeit 0 x
Test-Modus
Eject unterdrücken

Arbeitsoberfläche
Ordner erstellen Laufwerk Z:
Fehleranzeige automatisch starten

Quellpfad A:\INSTALL.EXE
Zielpfad C:\CDWFS

(c) Copyright 1998 RSJ Software GmbH
Industriestr. 10, 82110 Germering
Internet http://www.rsj.de

Installation beginnen... | Deinstallation... | Aktuelle Informationen... | Abbruch

tet wird, nicht genug Platz vorhanden sein, wird der Kopiervorgang abgebrochen, und unterschiedlichste Effekte bis zum Systemstillstand können auftreten.

❹ Starten Sie das System nach Beendigung des Installationsprogrammes neu.

Beachten Sie außerdem die Änderungen, die das *RSJ-Installationsprogramm* in Ihrer CONFIG.SYS vornimmt. Wenn Sie nur einen SCSI-Recorder verwenden, kommentieren Sie die Anweisung:

BASEDEV=RSJIDECD.FLT

mit einem REM aus. Den IDE-Filtertreiber braucht man in diesem Fall gar nicht. Er wird jedoch standardmäßig in das System eingebunden.
Das Programm nimmt noch weitere Veränderungen in der CONFIG.SYS vor, das sind im einzelnen:

BASEDEV=RSJIDECD.FLT
RSJIDECD-Filtertreiber, der den IBM-Treiber IBMIDECD.FLT ersetzt. Die BASEDEV-Anweisung, die IBMIDECD.FLT installiert, wird vom Installationsprogramm auskommentiert (mögliche Probleme s.u.).

BASEDEV=LOCKCDR.FLT
LOCK-Treiber, der CD-Recorder beim Systemstart erkennt und für den OS/2-Einheitenmanager sperrt. Dadurch stehen die Recorder erst nach Anmeldung beim CD-Writer-Dateisystem zur Verfügung.

DEVICE=C:\CDWFS\RSJSCSI.SYS
RSJ-SCSI-Treiber, der für den Betrieb von SCSI-CD-Recordern notwendig ist.

IFS=C:\CDWFS\CDWFS.IFS
Das installierbare RSJ-CD-Writer-Dateisystem.

RUN=C:\CDWFS\CDWFSD.EXE
Anweisung, die den CD-Writer-Hintergrundprozeß startet. Dieser Prozeß ist für das Schreiben von Daten auf die CD verantwortlich. Das Dateisystem leitet Befehle an diesen Dämonprozeß weiter.

Die Parameter des Programms CDWFSD.EXE finden Sie auch in den Einstellungsfeldern des Installationsprogramms wieder. Daneben trägt das Installationsprogramm den CDWFS-Pfad auch in den **PATH** und **LIBPATH** des Systems ein. Zusätzlich wird die Anweisung

BASEDEV=OS2ASPI.DMD

in die CONFIG.SYS eingetragen, sollte sie noch nicht vorhanden sein. Dieser Treiber ist übrigens auch für den Betrieb von *CDRecord/2* notwendig.

Die Dateien *rsjidecd.flt*, *rsjidecd.sym* und *lockcdr.flt* finden Sie nach der Installation im Stammverzeichnis des Installationslaufwerkes. Sollten sie dort stören, verschieben Sie sie in das Verzeichnis \OS2\BOOT.

Probleme nach der Installation
Schwierigkeiten während der Installation traten nicht auf, bis auf die Tatsache, daß das Installationsprogramm die *RSJ-WPS-Klassen* auf beiden Testrechnern manchmal nicht registrieren konnte (Fehlermeldungen s. *Abb.* 2). Das beeinträchtigte aber in keiner Weise die Funktionalität des Programmpaketes bis auf das Fehlen eines CD-Recorderobjektes nach Anmelden des Laufwerkes, so daß man diese Meldungen (die das Installationsprogramm selbst nur als Hinweise deklariert) eigentlich ignorieren kann - wenn man nicht mit der WPS arbeitet. Das wird aber nicht sehr oft vorkommen, und wenn man zum Aufheben des Schreibschutzes oder dem Löschen einer CD-RW auf die Kom-

mandozeile zurückgreifen muß, ist das natürlich schade. Wichtiger jedoch ist ein einwandfreies Arbeiten des Dateisystems, und das ist nicht zu beanstanden. Das Updateprogramm auf die Version 2.74 des *RSJ CD-Writers* registrierte die WPS-Klassen im übrigen nachträglich korrekt, so daß in späteren Versionen des Programms diese Fehler nicht mehr auftreten sollten.

Ein weitaus schwerwiegenderes Problem stellte sich jedoch auf dem Rechner mit dem HP-Gerät, also dem IDE-Recorder heraus: Hier blieb das System nach dem Laden der *RSJ*-Treiber mit einem **Trap** 000d hängen. RSJ macht allerdings in der Readme-Datei der Installation auf dieses Problem aufmerksam. Der Grund hierfür liegt in einer speziellen Variante des IBMIDECD.FLT-Filtertreibers namens *RSJIDECD.FLT*, der »Schreibkommandos und Kommando-Timeouts korrekt bearbeitet«. Weiter wird darauf hingewiesen, daß »auf schnellen Systemen [...] RSJIDECD.FLT Probleme verursacht« Zur Abhilfe soll man den IBM-IDECD-Filtertreiber wieder aktivieren.

Der Trap tritt übrigens nicht direkt beim Laden des *RSJ-IDECD-Treibers* auf, sondern erst dann, wenn der OS/2-Einheitenmanager für CD-ROM-Laufwerke, OS2CDROM.DMD, geladen wird. Sollte bei Ihnen ebenfalls das zuvor beschriebene Problem auftreten, gehen Sie wie folgt vor:

❶Öffnen Sie die CONFIG.SYS mit einem Texteditor, nachdem Sie das System mit Disketten, einer Wartungspartition oder einer anderen Betriebssystempartition gestartet haben.

❷Suchen Sie die Anweisung:

REM BASEDEV=IBMIDECD.FLT

welche das Installationsprogramm auskommentiert hat und entfernen Sie das REM.

❸Suchen Sie die Anweisung:

BASEDEV=RSJIDECD.FLT

und kommentieren Sie diese mit einem REM aus, oder löschen Sie die Zeile.

Anschließend startete unser System wieder. Übrigens traten beim Arbeiten mit dem IDE-Laufwerk auch ohne den *RSJIDECD-Filtertreiber* keine Schwierigkeiten auf, auch nicht nach längeren und umfangreicheren Sitzungen, in denen wir mehrere CDs nacheinander brannten, vom CD-ROM-Laufwerk auf den Recorder kopierten usw.

Interessant ist auch der Fixpaklevel des Betriebssystems in Zusammenhang mit dem oben beschriebenen Trap: Auf einem Warp 3-System trat er nur sporadisch auf, nach dem Aufspielen des FP 38 allerdings gar nicht mehr. Auf dem Warp 4-System konnten wir den Trap hingegen immer beobachten, egal

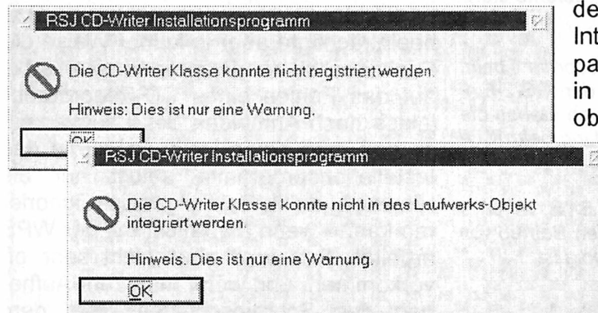


Abb.2: Mögliche Fehlermeldungen während der Installation des RSJ CD-Writers

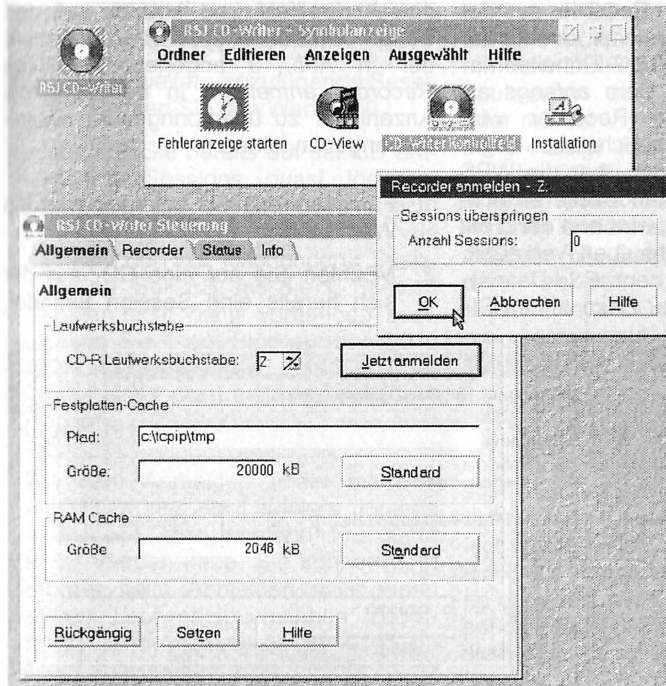


Abb. 3: Den CD-Recorder über die RSJ CD-Writer Steuerung anmelden

ob wir mit einem ungefixten oder mit verschiedenen Fixpaks korrigierten Warp arbeiteten. Wir nehmen jedoch an, daß die Hardwarekonfiguration weitaus entscheidender für das Auftreten des Traps ist. Sie müssen daher von Fall zu Fall entscheiden, ob ein Austrag des *RSJ-Filtertreibers* nötig ist, sofern Sie einen IDE-Recorder verwenden. Erschrecken Sie sich also nicht, falls Ihr System nach der Installation des CD Writers abstürzt, das Problem läßt sich leicht beseitigen.

Trotz allem ist dieser Fehler peinlich. Wenn man schon einen speziellen Filtertreiber entwickelt, sollte man auch darauf achten, daß ein Systemabsturz vermieden wird, ob das System nun schnell ist oder nicht. Der *RSJ-Filtertreiber* sollte also noch einmal einem Review unterzogen werden.

Das Beschreiben von CDs

Nach der erfolgreichen Installation des *CD-Writers* und dem Neustart des Systems findet man auf der Arbeitsoberfläche den Ordner *RSJ CD-Writer*, in dem die Objekte *CD-Writer Kontrollfeld*, *Fehleranzeige starten*, *CD-View* und *Installation* vorhanden sind. Mit dem Objekt *Installation* kann das Programmpaket wieder vom System entfernt werden. Die *Fehleranzeige* wird automatisch gestartet. Möchte man das nicht, lösche man die im Ordner *Systemstart* erzeugte Referenz. Das *CD-Writer Kontrollfeld* ist das wichtigste der Objekte, das sich als Notizbuch präsentiert. Wer nicht mit der WPS arbeitet, kann das Kontrollfeld über die Kommandozeile durch Eingabe des Befehls *cdwctl* starten. *Abbildung 3* zeigt neben dem geöffneten Kontrollfeld auch den *RSJ*-Ordner und das kleine Fenster zum Anmelden des Laufwerkes. Dazu sollte man auch auf die Meldungen des LOCKCDR-Treibers beim Systemstart achten: Der Treiber erkennt die im System installierten CD-ROM-Laufwerke und Recorder, ignoriert die CD-ROM-Laufwerke und

entfernt die erkannten Recorder aus der Laufwerksliste des Systems, um sie der Kontrolle durch den OS/2-Einheitenmanager zu entziehen. Dem anfangs als normal zugänglichen Recorder wird dann kein Laufwerksbuchstabe zugeordnet, womit er weder über die WPS noch über die Kommandozeile erreichbar ist. Das Laufwerk wird erst dann mit einem Laufwerksbuchstaben verknüpft, wenn der Recorder dem *RSJ-Dateisystem* gemeldet wurde. Dazu öffnen Sie das Kontrollfeld und klicken auf den Button *Jetzt anmelden*. Dann erscheint der in *Abb. 3* angegebene Dialog *Recorder anmelden*, in dem Sie die Anzahl der zu überspringenden Sessions angeben können:

1. Für eine neue CD belassen Sie den Wert auf 0.
2. Möchten Sie auf eine CD, auf der sich bereits eine Session befindet,

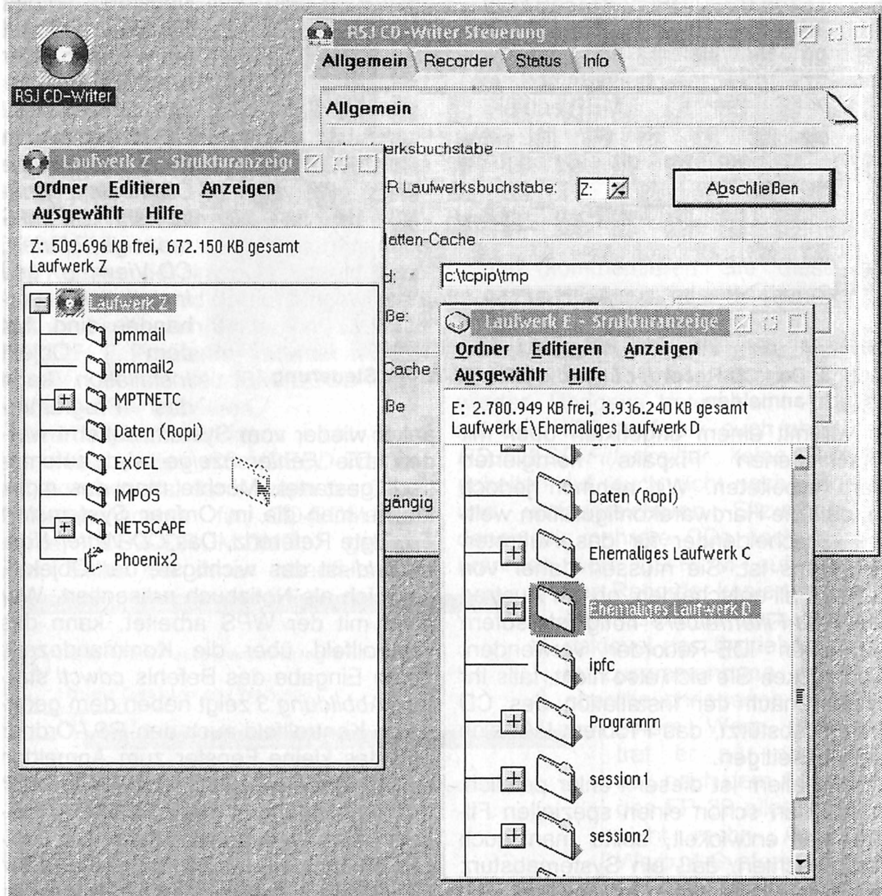


Abb. 4: Nachdem der Recorder am RSJ-Dateisystem angemeldet wurde, kann man ihn wie ein normales Laufwerk benutzen, mit der WPS und der Kommandozeile

eine zweite Session hinzufügen, belassen Sie den Wert auf 0.

3. Für jede weitere anzufügende Session belassen Sie den Wert auf 0.

4. Möchten Sie bereits auf der CD enthaltene Sessions quasi löschen, etwa weil Sie die Daten darin nicht mehr brauchen, geben Sie die Anzahl der »zu löschenden« Sessions an. Soll z.B. nur die 1. Session später nicht mehr vorhanden sein, geben Sie 1 an. Sind auf der CD 4 Sessions vorhanden, und Sie benötigen alle 4 nicht mehr, geben Sie 4 an usw.

Nach dem Anmelden findet man im Laufwerksordner ein Laufwerk namens z:, das wie eine normale Festplatte benutzt werden kann. Sie können dem Recorder beim Anmelden auch einen anderen Laufwerksbuchstaben zuordnen (z: ist die Standardeinstellung, da dieser Laufwerksbuchstabe in der Regel noch nicht vergeben ist). Übrigens können Sie den Recorder auch über die Kommandozeile dem Dateisystem melden. Dazu dient der Befehl *cdattach*, dem der Laufwerksbuchstabe übergeben wird, den der Recorder später haben soll:

```
cdattach z:
```

Die Anzahl der zu überspringenden Sessions ist dabei 0. Sie können diesen Wert durch den Parameter -l (klein L) verändern. Möchten Sie z.B. die 1. Session überspringen, geben Sie ein:

```
cdattach z: -l1
```

Ist der Anmeldevorgang abgeschlossen, kann man auf dem neuen Laufwerk kopieren, verschieben, löschen und umbenennen, womit das Beschreiben einer CD für den Benutzer völlig transparent durchgeführt wird, da man mit dem CD-Recorder wie mit anderen Laufwerken über die Kommandozeile oder mit der WPS arbeiten kann. *Abbildung 4* zeigt das Kopieren von Daten auf die CD-R mit Hilfe der WPS. Das Anlegen eines Images vor dem Brennvorgang entfällt also, und der Umgang mit dem *RSJ-Dateisystem* gestaltet sich damit als sehr komfortabel. Möchten Sie komplette CDs kopieren, greifen Sie am besten auf die Kommandozeile zurück. Mit dem Befehl:

```
xcopy f:\* z:\ /s /e /r /v /o /t /h
```

schreiben Sie den Inhalt einer CD, die sich im CD-ROM-Laufwerk f: befindet, auf die CD-R, die im Recorder z: eingelegt wurde. Wie Sie sehen, können Sie auf diese Weise mit einigen Batchdateien oder einem kleinen REXX-Programm sehr einfach Backups durchführen und sich teure Lösungen mit speziellen Programmen oder Streamern schenken. Bänder weisen ohnehin keine große Datensicherheit auf, so daß die CD-R als v.a. günstigeres Backupmedium ebenfalls in Frage kommt. Durch die einfache Entwicklung Ihres eigenen Backupprogrammes können Sie eine auf Ihr System optimal abgestimmte Lösung entwerfen. Auch für den geschäftlichen Einsatz kommt also die Anschaffung eines CD-Recorders und der entsprechenden Software in Frage und sei Ihnen empfohlen.

Nachdem alle Daten auf die CD geschrieben worden sind, schließt man die aktuelle Session ab, wodurch das Laufwerk abgemeldet wird. Am komfortabelsten geht das über die *CD-Writer* Steuerung, indem Sie auf den Button *Abschließen* drücken. Daraufhin erscheint ein kleiner Dialog, mit dem Sie angeben können, ob Sie die Session ohne oder mit Schreibschutz abschlie-

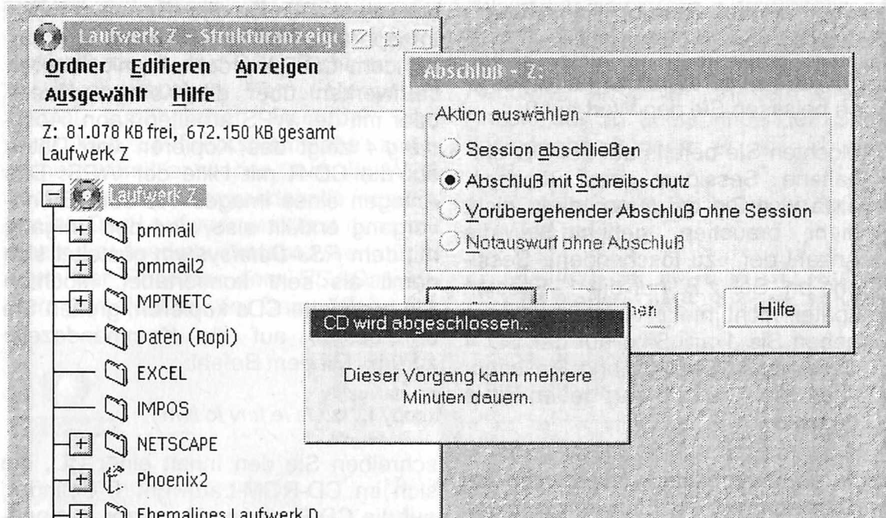


Abb. 5: Nachdem alle Daten auf die CD kopiert wurden, läßt sich die CD mit unterschiedchen Optionen abschließen.

ßen möchten (*Session abschließen, Abschluß mit Schreibschutz*) oder ob Sie die CD vorübergehend abschließen wollen, ohne die Session zu schließen (*Vorübergehender Abschluß ohne Session*). In allen Fällen werden die Daten des Schreibpuffers auf die CD geschrieben. Im zweiten Fall können Sie die CD ausgeben und später weitere Tracks der noch offenen Session hinzufügen. Die CD ist in diesem Zustand natürlich noch nicht von normalen CD-ROM-Laufwerken lesbar. Mit der letzten Option (*Notauswurf ohne Abschluß*) wird die CD sofort ausgegeben, ohne daß die Daten des Schreibpuffers auf die CD geschrieben werden. Wenden Sie diese Option am besten nur dann an, wenn Sie noch keine Daten auf die CD geschrieben haben oder wenn die CD beschädigt wurde. Sie können die Session auch über die Kommandozeile abschließen:

cdattach z: -<parameter>

wobei -<Parameter> folgende Werte annehmen kann:

-s
Entspricht: Session abschließen
-x
Entspricht: Abschluß mit Schreibschutz
-c
Entspricht: Vorübergehender Abschluß ohne Session
-no_flush
Entspricht: Notauswurf ohne Abschluß

Der Abschluß einer Session dauert etwa 5 bis 7 Minuten. Danach ist der Recorder wieder aus dem System verschwunden.

Anmelden ohne den LOCK-Treiber?

Mit dem LOCKCDR.FLT-Treiber von *RSJ* schleichen sich einige Nachteile auf dem System ein, die zu folgenden Fragen führen: Wie verwaltet man mehrere Recorder, wenn die Laufwerksbuchstaben nicht mehr vorhanden sind? Und wie kann man den Recorder noch

als normales CD-ROM-Laufwerk nutzen? Bevor man hier ständig am An- und Abmelden ist, um den Recorder je nach Einsatzzweck in das System einzubinden, kann man den *RSJ CD-Writer* auch ohne den LOCK-Treiber benutzen, vorausgesetzt, man verwendet SCSI-Recorder. Verfahren Sie dazu wie folgt:

❶ Entfernen Sie die Anweisung

BASEDEV=LOCKCDR.FLT

aus der CONFIG.SYS, führen Sie einen Systemabschluß durch, und starten Sie das System neu.

❷ Sie finden nach dem Neustart dann einen Laufwerksbuchstaben für Ihren Recorder vor (oder mehrere Buchstaben für mehrere Recorder), die Sie im Laufwerksordner als CD-ROM-Symbole finden. Die Recorder lassen sich dann ganz normal als CD-ROM-Laufwerke benutzen

❸ Um auf einen Recorder zu schreiben, melden Sie ihn dem *RSJ CD-Writer* Dateisystem an. Dazu öffnen Sie eine Kommandozeile und geben den Befehl *cdattach* mit folgender Syntax ein:

cdattach <LW> -d cdr:<adapter.device>

Dabei steht LW für Recorder-Laufwerk und bezeichnet den Laufwerksbuchstaben, den Ihr Recorder nach der Anmeldung erhalten soll. Der Parameter -d dient zur Angabe der Identität des Recorders auf dem SCSI-Bus. Die Bezeichnung des Recorders wird dabei dem Argument *cdr:* in der Form *adapter.device* übergeben, wobei *adapter* die Nummer Ihres Hostadapters und *device* die ID Ihres Recorders auf dem SCSI-Bus des zuvor angegebenen Adapters ist. Wenn Sie z.B. am ersten SCSI-Hostadapter einen CD-Recorder mit der ID 4 angeschlossen haben, sieht der Befehl zum Anmelden wie folgt aus:

cdattach z: -d cdr:0.4

Achten Sie darauf, *adapter* und *device* durch einen Punkt zu trennen.

❹ Melden Sie weitere Recorder auf diese Weise an. Sie müssen dem zweiten Recorder nur einen anderen Laufwerksbuchstaben zuweisen, z.B. y:

❺ Arbeiten Sie mit den Recordern wie gewohnt, und melden Sie sie ebenfalls wie sonst mit dem Befehl *cdattach* oder mittels des *CD-Writer Kontrollfeldes* ab (s.o.).

Die Recorder, die Sie nicht zum Aufzeichnen verwenden, können Sie als CD-ROM-Laufwerke benutzen. Diese Vorgehensweise funktioniert, jedoch findet man nichts darüber im Handbuch.

Tip: *Schreiben Sie für jeden Recorder eine kleine Batchdatei, welche die Anmeldung übernimmt, und benennen Sie die Datei mit dem Herstellernamen Ihres Recorders, z.B.* yamaha.cmd, *damit Sie den gesamten Befehl nicht immer wieder eingeben müssen.*

Auch CD-ROM-Laufwerke lassen sich dem *RSJ-Dateisystem* melden. Dazu verwendet man ebenfalls den Befehl cdattach:

cdattach z: -d f:

Damit wird das CD-ROM-Laufwerk mit dem Buchstaben f: dem Dateisystem als Laufwerk z: gemeldet.

Das ist immer dann praktisch, wenn man auch unter OS/2 CDs lesen möchte, deren Verzeichnisinformationen die *Rockridge-Extensions* beinhalten.

! *Melden Sie auf diese Weise Ihren Recorder nicht an, wenn Sie vorhaben, auf eine CD-R zu schreiben. Der Zugriff auf den Datenträger würde dann verweigert.*

Zusatzinformationen zum Dateisystem
Im folgenden möchten wir Ihnen noch einige Informationen zum Dateisystem und den Einstellungsmöglichkeiten über die WPS und der Kommandozeile geben.

Abbildung 6 zeigt das Kontextmenü des Laufwerkobjektes, das nach der Anmeldung des CD-Recorders im Ordner Laufwerke verfügbar ist. Über dieses Menü können Sie den Schreibschutz einer CD-R aufheben, die CD abschließen und eine CD-RW löschen. Wenn Sie lieber mit der Kommandozeile oder selbst geschriebenen Programmen arbeiten, die auf die Kommandozeile zurückgreifen, finden Sie im folgenden

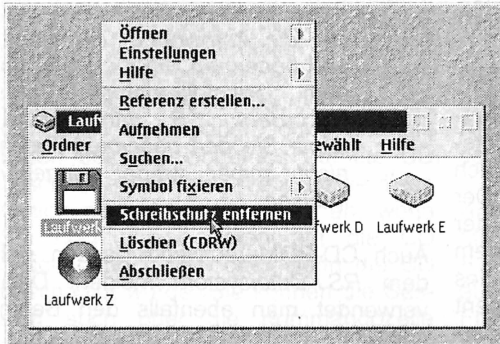


Abb. 6: Das Kontextmenü des Recorder-Laufwerkobjektes der WPS nach der Anmeldung des CD-Recorders am RSJ-Dateisystem.

die Befehle für die entsprechenden WPS-Funktionen. Sie können dann auch von einer Wartungspartion aus, auf der nur ein Textmodus-OS/2 zur Verfügung steht, CD-Rs beschreiben. Um den Schreibschutz einer CD-R aufzuheben, geben Sie ein:

format <LW> /unseal

Dabei ist <LW> das Laufwerk Ihres Recorders nach der Anmeldung. Mit dem Parameter */unseal* heben Sie den Schreibschutz auf.

Mit dem Befehl *format* löschen Sie auch eine CD-RW. Geben Sie dazu den Befehl mit dem Parameter */erase* ein:

format <LW> /erase

Dabei ist <LW> wieder der Laufwerksbuchstabe des CD-Recorders.

Abbildung 7 zeigt die Seite *Recorder* der *CD-Writer Steuerung*. Sie können die hier angegebenen Einstellungen auch über die Kommandozeile festlegen. Um die Schreibgeschwindigkeit für einen Recorder zu setzen, benutzen Sie den Befehl *cdspeed*:

cdspeed <LW> -s <Geschw.> [-e] [-f]

<LW> ist dabei wieder der Buchstabe des angemeldeten CD-Recorders. Mit dem Parameter -s setzen Sie den Geschwindigkeitsfaktor, also 1 für Singlespeed, 2 für Doublespeed usw. Um z.B. den Recorder z: auf Quadspeed zu setzen, geben Sie ein:

cdspeed z: -s 4

Mit dem Parameter -e können Sie alle Operationen mit dem Recorder im Testmodus durchführen. Er ist optional und eignet sich, um die richtige Schreibgeschwindig-

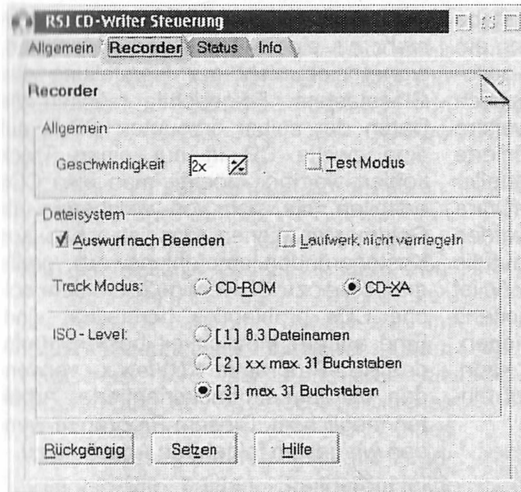


Abb. 7: Einstellungen des Dateisystems festlegen

keit in Relation zum Systemdurchsatz zu finden. Der Parameter -f dient zur Angabe des Dateisystems, an das der Recorder gemeldet wurde. Da der Standardwert dieses Parameters CDWFS ist, können Sie diese Option fortlassen.

Die Parameter zum ISO-Level und Trackmodus können Sie auch in der CONFIG.SYS dem Hintergrundprozeß CDWFSD.EXE übergeben. Wir erwähnen hier nur die Parameter, die angegeben sein müssen, und die Sie bei Bedarf verändern sollten:

-p <Cachepfad>
Pfad, in dem das Dateisystem die Cachedateien ablegt. Im CD-Writer Notizbuch auf der Seite Allgemein unter Festplattenchache-Pfad.

-c <Größe>
Platz in KByte der durch Cachedaten im durch -p angegebenen Verzeichnis belegt wird. Standardgröße: 20000. Im Notizbuch: Festplattenchache-Größe.

-b <Größe>
Bestimmt die Größe des RAM-Puffers in KByte. Standardgröße: 2048. Im Notizbuch finden Sie die Einstellung unter RAM-Cache - Größe.

-t <Modus>
Legt den Trackmodus fest. Gültige Werte für <Modus> sind 1 für normale Datentracks und 2 für XA-Tracks. Standardwert: 2. Im Notizbuch unter Dateisystem-Track Modus.

-i <Level>
Gibt den ISO-Level für das Dateisystem der CD fest (s. auch Kapitel: Die CD-ROM). Gültige Werte sind 1, 2 und 3, wobei jede Ziffer für den jeweiligen Level steht. Standardwert: 3. Im Notizbuch unter ISO-Level.

Mit der CD-Writer Steuerung ist es auch möglich, Statusinformationen zur beschriebenen CD abzufragen. *Abbildung 8* stellt die Seite *Status* dar. Klicken Sie auf den Button *CD Status* dieser Seite, so wird die Belegung des Datenträgers als Tortendiagramm angezeigt. Die gleichen Informationen erhalten Sie allerdings auch, indem Sie den Befehl *chkdsk* eingeben:

chkdsk <LW> [/v]

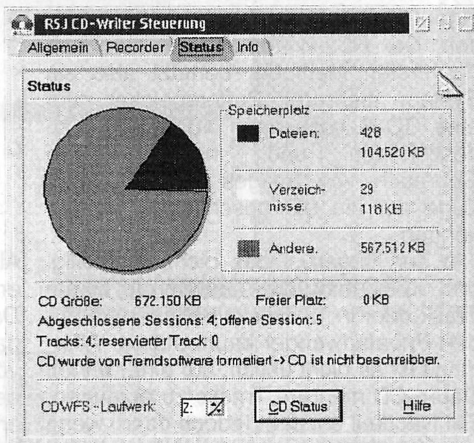


Abb. 8: Statusfunktionen über die CD abfragen

Dabei ist <LW> wieder der Buchstabe des Recorders nach der Anmeldung am Dateisystem. Der Parameter /v ist optional und gibt neben den im Notizbuch dargestellten Informationen weitere durch die Norm ISO 9660 definierte Informationen aus, etwa der Hersteller der CD, die dazu verwendete Software usw. Diese Zusatzinformationen werden beim Abschluß einer Session mit auf die CD geschrieben. Der Parameter /v ist im *RSJ-Handbuch* hinreichend beschrieben, ebenso die Informationen, die man mit auf die CD schreiben kann, so daß wir hier nicht näher darauf eingehen müssen.

Die beschriebenen CDs

Die mit dem *RSJ CD-Writer* beschriebenen CDs bestehen immer aus mehreren Tracks, im Durchschnitt 14 bis 18, wenn man die gesamte CD beschreibt. Mit dem Dateisystem ist es völlig unmöglich, eine CD zu erzeugen, auf der sich alle Daten in nur einem Track befinden. Das ist logisch, wenn man bedenkt, wie der *RSJ-Writer* Daten auf eine CD-R kopiert: Bereits im CD-Recorder-Test haben wir auf die beiden Möglichkeiten, eine CD-R zu beschreiben, hingewiesen. Der *RSJ-Writer* arbeitet nur nach der ersten Methode, benutzt also ein System mehrerer Pufferspeicher, um eine CD-R sozusagen »on-the-fly« zu beschreiben und macht keinen Gebrauch von Images, die in einem Zuge auf die CD geschrieben werden können.

In der Regel ist das nicht nachteilig, und ob sich nun die Daten in nur einem Track oder in 18 Tracks befinden, dürfte den Privatanwender kaum stören. Allerdings ist es nicht üblich, auf einer reinen Daten-CD mehrere Tracks zu erzeugen. Ein Nachteil entsteht jedoch dann, wenn man einen Goldmaster erstellen möchte, um die CD in Serie produzieren zu lassen. Eine CD, auf der die Daten in mehr als einem Track vorhanden sind, eignet sich nicht zur Herstellung des Glasmasters. Folgerichtig müssen die Daten der selbst erzeugten CD-R auf eine zweite CD in nur einen Track kopiert werden. Möchte man also CDs erstellen, die sich von vornherein als Goldmaster eignen (das ist schon aus Kostengründen empfehlenswert), greife man lieber zu CDRecord/2, mit dem solche CDs problemlos herstellbar sind. Eine andere Möglichkeit besteht darin, das *RSJ*-Programm *CDView* zu verwenden, mit dem ein trackorientiertes Arbeiten möglich ist. Diesem Programm wenden wir uns im folgenden Abschnitt zu.

CDView

Das Programm *CDView* ermöglicht im Gegensatz zum *RSJ-Dateisystem* ein trackorientiertes Arbeiten mit CDs. Die besonderen Vorteile dieses Programms lassen sich wie folgt zusammenfassen:

- Mehrere CD-Recorder lassen sich gemeinsam nutzen.
- Das Anmelden eines Laufwerkes ist nicht erforderlich.
- Man kann CD-Rs erstellen, auf denen sich die Daten in nur einem Track befinden.
- Das Kopieren von CDs ist mit dem Programm via Drag & Drop sehr einfach.

Als Nachteil kann sich die Tatsache erweisen, daß *CDView* und das *RSJ-Dateisystem* nicht zusammenarbeiten: das eine Produkt ist mit dem anderen völlig inkompatibel. Man kann mit den beiden Programmen auch nicht parallel arbeiten. Welches von beiden man benutzt, hängt demnach vom Einsatzzweck ab. Um Backups anzufertigen,

eignet sich das Dateisystem auf jeden Fall besser, weil man sehr intuitiv damit arbeiten kann. Für die Herstellung von Goldmastern oder dem Kopieren von Audio-CDs sollte man auf *CDView* zurückgreifen.

CDView ist sehr einfach zu benutzen. Im *RSJ*-Ordner auf der WPS findet man das Objekt CD-View. Hinter ihm verbirgt sich ein weiterer Ordner, in dem zwei Programmobjekte liegen, welche sich auf das Programm *cdview.exe* im \CDWFS-Verzeichnis beziehen. Dem Programm wird als Parameter die Bezeichnung des CD-Recorders, eines CD-ROM-Laufwerkes oder auch eines Verzeichnisses auf der Festplatte zugewiesen. Die beiden Standardobjekte beziehen sich auf ein Verzeichnis auf der Festplatte, in das man einzelne Tracks zwischenspeichern kann, und auf den im System vorhandenen CD-Recorder.

Sie können sich für einen zweiten oder weitere Recorder und auch für CD-ROM-Laufwerke weitere Programmobjekte anlegen oder *cdview* direkt über die Kommandozeile starten, wobei Sie das entsprechende Laufwerk dem Programm als Parameter übergeben. Möchten Sie ein CD-ROM-Laufwerk mit *CDView* starten, geben Sie ein:

```
cdview f:
```

Damit wird das CD-ROM-Laufwerk mit der Bezeichnung *f:* einer Programminstanz von *cdwiev* zugewiesen. Möchten Sie mit mehr als einem Recorder unter cdview arbeiten, geben Sie die Nummer des SCSI-Adapters und die ID des Recorders am SCSI-Bus an, z.B.:

```
cdview cdr:0.3
```

Auf diese Weise starten Sie eine Programminstanz von *cdview* und verknüpfen die mit dem CD-Recorder, der die SCSI-ID 3 am 1. Adapter im System hat.

Haben Sie eine Programminstanz von *CDView* gestartet, erscheint ein Programmfenster wie in *Abb. 9*. Das Programm liest beim Start die Trackinfor-

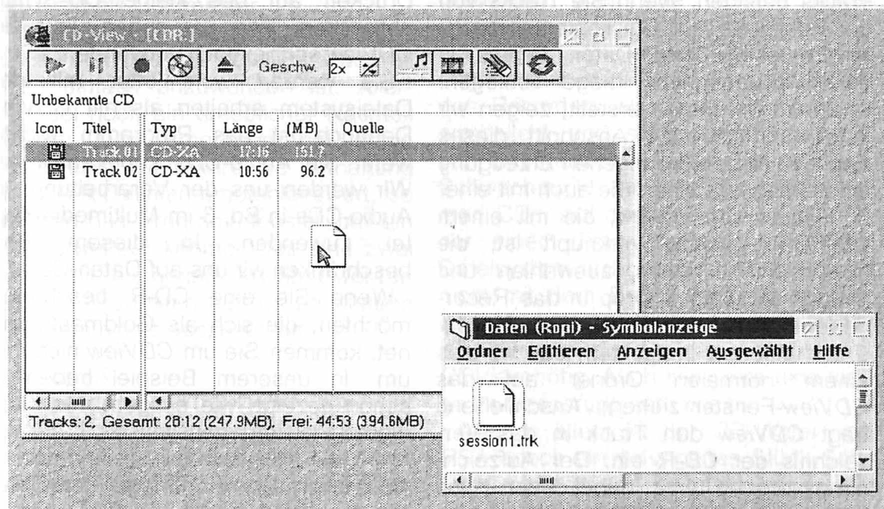


Abb. 9: Tracks von der Festplatte mit CDView kopieren.

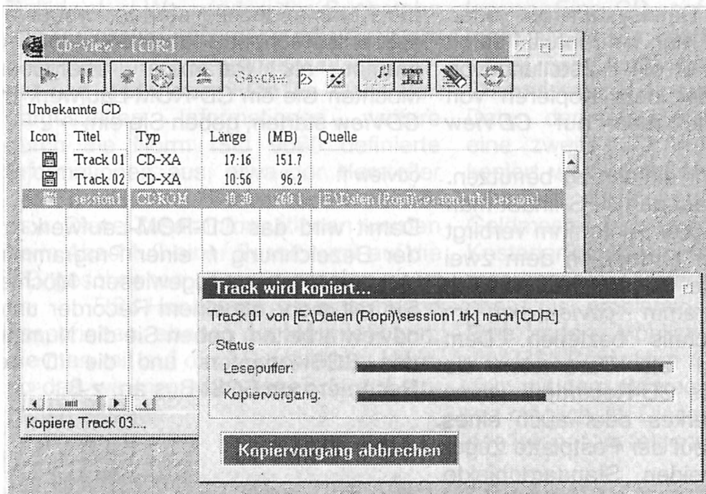


Abb. 10: Den Aufzeichnungsvorgang mit CDView starten

mationen von der CD und stellt sie übersichtlich in einem Container dar. In unserem Beispiel haben wir nur das CD-Recorder-Objekt im *CD-View*-Ordner geöffnet, um einen mit *mkisofs* vorbereiteten Track auf eine CD-R zu schreiben, auf der sich schon zwei Tracks befinden. Wenn Sie Tracks von der Festplatte auf CD kopieren möchten, müssen diese Daten bereits in einem sogenannten Image vorliegen. Wie man ein Image erstellt, zeigen wir Ihnen im nächsten Abschnitt dieses Kapitels. Neben der eigenen Erzeugung eines Images können Sie auch mit einer 2. Instanz von cdview, die mit einem CD-ROM-Laufwerk verknüpft ist, die gewünschten Tracks auswählen und einfach via Drag & Drop in das Recorderfenster ziehen (so wie wir es in *Abb. 9* mit unserem Image tun, das wir aus einem normalen Ordner auf das *CDView*-Fenster ziehen). Anschließend trägt *CDView* den Track in das Verzeichnis der CD-R ein. Der Aufzeichnungsvorgang wird dabei aber noch nicht gestartet, d.h. Sie können zunächst weitere Tracks auf das Programmfenster ziehen und sie dann ganz nach Belieben arrangieren. Erst dann wenn Sie auf den *Aufzeichnungsknopf* drücken (in der Abbildung der 3. von links), beginnt das CDView, den Track auf die CD zu kopieren. Der Vorgang wird durch ein Statusfenster dargestellt. Die Aufzeichnung sollten Sie nicht abbrechen, selbst wenn das Statusfenster einen entsprechenden Buttons bietet. Nach dem Aufzeichnungsvorgang können Sie weitere Tracks auf die CD kopieren oder die Session durch Drücken auf das kleine *CD-Symbol* abbrechen. Wie Sie sehen, läßt sich *CDView* sehr einfach verwenden.

Sie werden wohl öfter mit dem Dateisystem arbeiten als mit *CDView*. Dennoch ist das Programm wichtig, wenn Sie Audio-CDs kopieren wollen. Wir werden uns der Verarbeitung von Audio-CDs in Bd. 3 im Multimedia-Kapitel zuwenden. In diesem Band beschränken wir uns auf Daten-CDs.

Wenn Sie eine CD-R beschreiben möchten, die sich als Goldmaster eignet, kommen Sie um *CDView* nicht herum. In unserem Beispiel haben wir schon gezeigt, wie es geht. Hier noch einmal die Vorgehensweise:

❶ Erstellen Sie ein Image der Daten, welche auf die CD geschrieben wer-

den sollen (z.B. mit mkisofs). Bei der Betrachtung von cdrecord im nächsten Abschnitt dieses Kapitels erhalten Sie eine Schritt-für-Schritt-Anleitung zur Erstellung von Images. Achten Sie beim Erzeugen des Images darauf, daß die Datei die Erweiterung *.trk besitzt.

❶ Legen Sie eine neue CD-R in Ihren Recorder.

❶ Starten Sie dann *CDView* und übergeben Sie dem Programm als Parameter die Bezeichnung des CD-Recorders.

❶ Öffnen Sie den Ordner, in dem sich das Image befindet und ziehen Sie die Datei auf das geöffnete CDView-Recorderfenster.

❶ Starten Sie den Aufzeichnungsvorgang wie beschrieben und schließen Sie danach die Session ab.

Die auf diese Weise erstellte CD eignet sich direkt als Goldmaster.

CDView hat uns gut gefallen, v.a. weil es so einfach anzuwenden ist. Allerdings sei noch ein erheblicher Nachteil erwähnt: Das Programm hat keine Online-Hilfe. Auch wäre die Einhaltung der SAA-Richtlinien empfehlenswert, so daß man das Programmfenster um ein Menü erweitern könnte. Das sind zwei Punkte, die als negativ erwähnt werden müssen. Ansonsten gibt es an diesem Programm nichts auszusetzen.

Zum Schluß

Uns hat der *RSJ CD-Writer* sehr gut gefallen. Das Programmpaket ist, zugegebenermaßen nicht billig, ab 400 Mark kann man eine *RSJ*-Version erstehen; aber die Software ist ihr Geld wert und der Preis auch noch verträglich. Zudem hat *RSJ* zugesichert, den *CD-Writer* für OS/2 ohne Einschränkung weiterzuentwickeln. Das gilt insbesondere für die Unterstützung neuer Recorder. Für sein Geld bekommt man auch einen guten Support, der an dieser Stelle lobenswert erwähnt werden sollte. Hat man ein Problem, ist Hilfe nicht fern. Anfragen über eMail werden schnell und v.a. kompetent beantwortet.

Allerdings wünscht man sich einige Verbesserungen am Programm. Die erste betrifft die Dokumentation. Ein besseres Handbuch oder ein umfangreiches INF-Dokument wären von großem Nutzen, v.a. weil tiefergehendere Fragen unbeantwortet bleiben. Besonders wichtige Informationen nur in eine README-Datei zu packen, wiegt ein gutes Online-Buch nicht auf. Auch die bereits erwähnten Mängel an CDView sollten ausgeglichen werden, da sie dem an sich guten Gesamtbild schaden. Weiterhin zeigt sich die Notwendigkeit zur Verbesserung der graphischen Schnittstelle: Eine umfangreichere CD-Steuerung oder eine bessere WPS-Unterstützung wäre hier von Vorteil. So wäre z.B. ein neues CD-Recorder-Objekt, das auch eine Verwendbarkeit des Recorders als normales Laufwerk ermöglicht und ständig verfügbar ist, von besonderem Vorteil. Ein weiterer Schwachpunkt ist die Tatsache, daß man CDs, auf denen sich die Daten in nur einem Track befinden, mit dem Dateisystem nicht erzeugen kann, auch nicht mit dem Befehl *cdcopy*, den wir deshalb auch nicht erwähnten. Daher ist Umkopieren oder das Arbeiten mit *CDView* nötig. Auch hier wäre eine kleine Verbesserung sehr positiv.

Alles in allem ist der *CD-Writer* von *RSJ* jedoch ein gelungenes Stück Software, schnell und kompakt entwickelt. Das Produkt bietet neben Benutzerkom-

fort auch ein einfach zu verwendendes API, so daß man eigene Oberflächen für das Dateisystem entwickeln kann. Das Programmpaket arbeitet OS/2-konform, belastet das System nur wenig und ist durch die Möglichkeit die Kommandozeile zu benutzen auch sehr flexibel. Audio-CDs kann man mit *CDView* bearbeiten bequem und sogar bootfähige CDs lassen sich mit diesem Produkt herstellen (s. Bd. 3), womit der *RSJ CD-Writer* ein breites Spektrum an Anwendungszwecken abdeckt. Letztlich bleibt also zu sagen, daß wir uns über die Software sehr gefreut haben und ihr ohne Bedenken eine Empfehlung aussprechen können.

Kontakt mit RSJ

Anschrift:
RSJ Software GmbH
Industriestr. 10
82110 Germering

Tel.: 089/894142-0
Fax: 089/894142-80

Internet: http://www.rsj.de
Support: support@rsj.de

Über das Internet erhält man neben einer Demoversion des CD-Writers auch Updates zum Programm. Ferner sind zum Download verfügbar:

- *o Dokumente im PDF-Format*
- *o FAQs*
- *o Dokumentation des API*

Eine OS/2-Newsgroup zum RSJ CD-Writer ist direkt über die Website zugänglich.

CDRecord/2

Neben dem zuvor vorgestellten Produkt von *RSJ* gibt es zum Beschreiben von CDs auch noch eine andere Lösung, die man unter OS/2 einsetzen kann, nämlich *CDRecord/2*. Dieses Programm ist ein Port des Unix-Programms *cdrecord* und eignet sich, ein wenig Einarbeitung vorausgesetzt, ebenfalls sehr gut für den Gebrauch unter Warp. Daneben ist es einfach erweiterbar und flexibel im Gebrauch. Dem *RSJ CD-Writer* steht es hinsichtlich der Funktionalität in nichts nach.

Was ist CDRecord/2?

CDRecord/2 ist ein kommandozeilenorientiertes Programm. Allerdings gibt es auch graphische Frontends für den PM und die WPS. Vor allem letzteres dürfte für viele OS/2-Anwender interessant sein; und da man ein Programm für die Kommandozeile auch einfach in seine eigenen Anwendungen integrieren kann, machte uns das Freewareprogramm besonders neugierig[9].

Es stellte sich übrigens heraus, daß man damit ganz vorzüglich arbeiten kann, v.a. wenn es darum geht auf einfache Weise bootfähige CDs zu erstellen oder Goldmaster, die man direkt zum Glasmastering verwenden kann. Das Programm arbeitet mit einer anderen Methode als das *RSJ*-Programm. Dadurch ist es flexibler; auf der anderen Seite ist eine so einfache Handhabung wie mit dem *RSJ-Dateisystem* (noch) nicht möglich.

Was braucht man zum Betrieb?

Die aktuelle Version von *cdrecord* ist 1.8a19, die wir testeten. Ein entspre-

9 Das Programm ist eine ideale Grundlage für eigene Anwendungen. Wir haben es in den objektorientierten *Xelia-Schreibtisch* integriert und können komfortabel CD-Rs beschreiben. Die *cdrecord/2*- und *mkisofs*-Unterstützung wird in der kostenlosen Demoversion von *Xelia* enthalten sein. Alle WPS-Programmierer seien aufgerufen, für die beiden Programme entsprechende Klassen zu entwerfen. Die Mühe dürfte sich lohnen.

chendes Archiv finden Sie auf unserer Homepage. Alleine dieses Programm reicht allerdings nicht aus, um CD-Rs zu beschreiben. Sie benötigen außerdem:

- *mkisofs.exe*, ein Programm, das ISO CD-Images erstellt, die *CDRecord/2* dann auf die CD schreibt.
- Den Treiber *aspirout.sys*, einen ASPI-Router, der Anwendungen, die auf Ring 3 arbeiten, ermöglicht, direkt mit dem ASPI-Treiber OS2ASPI.DMD zu kommunizieren.
- die *EMX Runtime-Bibliothek* in der Version 0.9c oder höher.

Das Programm *mkisofs* müßten Sie sich eigentlich nicht unbedingt extra besorgen, da der *CDRecord/2* Distribution eine OS/2-Version von *mkisofs* beiliegt. Durch einen Bug in *cdrecord1.8a19* arbeitet die beigelegte *mkisofs*-Version aber nicht korrekt mit *cdrecord* zusammen. Laden Sie also beim Download von *cdrecord* gleich *mkisofs1.12b4* mit. Wenn Sie die hier genannten Versionen von *mkisofs* und *cdrecord* verwenden, können Sie CDs auch »on-the-fly« beschreiben (mehr dazu weiter unten). Auch das Erstellen von Multisession-CDs funktioniert dann wie gewünscht.

Wir haben für Sie *cdrecord1.8a19*, *mkisofs1.12b4*, die aktuelle *EMX Runtime-Bibliotek (Version 0.9d)* und den *aspirout.sys* auf unserer Homepage bereitgelegt. Sie finden alle Dateien aber auch in den üblichen Softwarearchiven wie LEO oder Hobbes.

Daß es Schwierigkeiten mit verschiedenen Versionen von *mkisofs* und *cdrecord* geben kann liegt daran, daß die OS/2-Programme stets direkt vom Original-Sourcecode portiert werden. Da sowohl *cdrecord* als auch *mkisofs* beständig weiterentwickelt werden, können natürlich auch immer wieder Bugs auftreten, die ebenfalls in den Port übernommen werden, wodurch Inkompatibilitäten möglich sind. Beachten Sie jedoch, die in diesem Artikel vorgestellten Versionen zu verwenden, dürften keine Schwierigkeiten auftreten.

Die Installation

CDRecord/2 auf dem System einzurichten, ist ausgesprochen einfach. Gehen Sie wie folgt vor:

❶ Installieren Sie zunächst die *EMX Runtime Bibliothek*, indem Sie das ZIP-Archiv in das Stammverzeichnis eines Laufwerkes (am besten das Ihres Bootlaufwerkes) legen und dort entpacken. Tragen Sie anschließend den Pfad:

<Inst.-Lw.>:\EMX\DLL

in die LIBPATH-Anweisung und den Pfad:

<Inst.-Lw.>:\EMX\EXE

in die PATH-Anweisung in Ihrer CONFIG.SYS ein.

❷ Enpacken Sie die Datei mit dem ASPI-Router in einem temporären Verzeichnis, kopieren Sie die Datei *aspirout.sys* in das Verzeichnis \OS2\BOOT Ihres Bootlaufwerkes, und nehmen Sie folgende Eintragung in der CONFIG.SYS vor:

<Inst.-Lw.>:\OS2\BOOT\ASPIROUT.SYS

❸ Suchen Sie die Anweisung:

BASEDEV=OS2ASPI.DMD

in der Datei CONFIG.SYS und fügen Sie der Anweisung den Parameter /ALL hinzu. Sollte OS2ASPI.DMD noch nicht installiert sein, binden Sie

den Treiber mit der BASEDEV-Anweisung in Ihr System ein.

! ***Achten Sie darauf, daß der Parameter /ALL dem ASPI-Treiber übergeben wird, daß die entsprechende Zeile in der CONFIG.SYS also so aussieht:***

BASEDEV=OS2ASPI.DMD /ALL

Ansonsten erhält* cdrecord *keinen Zugriff auf den SCSI-Bus.

❹Installieren Sie dann *cdrecord*. Entpacken Sie dazu das ZIP-Archiv in einem möglichst übergeordneten Verzeichnis (z.B. \PROGRAMME). Daraufhin wird ein neues Verzeichnis namens:

cdrecord-1_18a19_os2

erzeugt. Sie finden darin die Datei *cdrecord.exe*. Kopieren Sie diese am besten in das Verzeichnis \OS\APPS.

❺Entpacken Sie das ZIP-Archiv von *mkisofs1.12b4* im gleichen übergeordneten Verzeichnis wie *cdrecord*. Sie finden dann ein Verzeichnis namens:

mkisofs-1.12b4

und darin die Datei *mkisofs.exe*. Kopieren Sie diese in das gleiche Verzeichnis wie *cdrecord*, am besten wieder in \OS2\APPS.

❻Führen Sie einen Systemabschluß durch, und starten Sie den Rechner neu.

Damit ist die Installation von *cdrecord* abgeschlossen. Mehr als diese beiden kleinen Programme brauchen Sie nicht, um CDs zu erstellen. Das ist recht erstaunlich, wenn man bedenkt, welches große Anwendungsspektrum mit den beiden binären Winzlingen abgedeckt wird, und wie gut die Ergebnisse sind, die man mit den Programmen erzielt.

Die Arbeitsweise

Es gibt zwei Methoden mit *cdrecord* zu arbeiten:

1. Man erstellt mit *mkisofs* ein oder mehrere Images der Daten, die man auf die CD brennen möchte, und kopiert diese anschließend mit *cdrecord* auf die CD.
2. Man brennt die Daten »on-the-fly« auf die CD. Dazu ruft man *mkisofs* mit entsprechenden Paramtern und *cdrecord* in einer Zeile auf.

Die erste Methode ist die sicherste. Man kann so in aller Ruhe seine Daten zusammenstellen, das Image erzeugen und von *cdrecord* in einen Track auf der CD ablegen lassen. Es ist dabei zu beachten, daß *cdrecord* keine Abbruchmöglichkeit mehr nach Beginn des Schreibvorganges bietet. Man sollte daher zuerst überprüfen, ob genug Speicherplatz zur Aufnahme des Images (oder eines weiteren Images bei Multisession-CDs) auf der CD-R vorhanden ist.

Möchte man ohne ein Image auskommen, sollte man laut Autor beachten, daß das zum Einsatz kommende System auch entsprechend leistungsfähig ist, damit der Datenstrom während des Schreibvorganges nicht abreißt. Wir haben jedoch CDs auf diese Weise mit dem bereits beschriebenen 486er-System unter normaler Systembelastung beschrieben und stießen während des Schreibvorganges auf keinerlei Pro-

bleme. Das beweist wieder einmal mehr, wie sehr sich OS/2 für die Bearbeitung von Vorgängen in Echtzeit eignet.

Parameter von cdrecord und mkisofs
Während *mkisofs* zur Herstellung eines ISO 9660 konformen Dateisystems dient, übernimmt *cdrecord* die eigentliche Aufzeichnungsarbeit. Beide Programme verfügen über eine Fülle von Parametern. Die wichtigsten werden wir Ihnen in unseren Anleitungen einige Seiten später ohnehin präsentieren und ihre Bedeutung erklären. Damit Sie aber nicht immer in die README-Dateien der beiden Programme sehen müssen (denn eine andere Dokumentation gibt es leider nicht), finden Sie im folgenden alle wichtigen Optionen aufgeführt und erklärt. Wir empfehlen Ihnen, die beiden Listen vor der weiteren Lektüre und der Arbeit mit den beiden Programmen aufmerksam durchzulesen.

Beachten Sie auch die allgemeine Syntax von *cdrecord*:

```
cdrecord [allg.Parameter] dev=x,y
                  [Trackparameter] track1 ... trackn
```

Mit dev=x,y geben Sie die Identität Ihres Recorders auf dem SCSI-Bus an. Die *Allgemeinen Parameter* nennen Sie vor der Angabe des Recorders. Daran anschließend geben Sie die Trackparameter an. Zum Schluß nennen Sie die Images der einzelnen Tracks.

Die grundsätzliche Syntax von *mkisofs* lautet:

```
mkisofs [Parameter] -o image.file
                        /quellenverzeichnis
```

Die in Parameter festgelegten Optionen bestimmen dabei den Typus des zu erzeugenden CD-Images, das in der durch -o angegebenen Datei abgelegt wird. Hier kann auch eine vollständige Pfadangabe angegeben werden. Die Imagedatei muß übrigens keine besondere Erweiterung haben (sinnvoll wären *.raw, *.img, *.trk). Wenn Sie Images für *RSJ CDView* erzeugen, **müssen** Sie die jedoch die *.trk-Endung wählen. Zum Schluß übergeben Sie *mkisofs* die Bezeichnung des Verzeichnisses, in dem sich die Quellendateien und -Verzeichnisse befinden.

! ***Wenn Sie cdrecord oder mkisofs Verzeichnisnamen übergeben, achten Sie darauf, Verzeichnisse nicht mit dem unter OS/2 üblichen Backslash \ , sondern mit dem unter Unix gebräuchlichen Forwardslash / voneinander zu trennen.***

Sie müssen übrigens keine besondere Reihenfolge einhalten, wenn Sie den beiden Programmen die einzelnen Parameter übergeben.

Die wichtigsten Parameter von cdrecord
In diesem Abschnitt finden Sie die wichtigsten Parameter von *cdrecord* und ihre Bedeutung:

Allgemeine Parameter

-version
Durch Angabe dieses Prameters gibt *cdrecord* die Versionsnummer aus und beendet sich anschließend.

-v
Durch -v werden weitere Statusinformationen angezeigt, wenn auf die CD geschrieben wird. Wir empfehlen, diesen Parameter immer anzugeben.

-dummy
Mit diesem Parameter kann man einen Schreibvorgang im Testmodus laufen lassen.

-multi
Dieser Prameter ist *notwendig*, um Multisession-CDs zu erzeugen. Tracks werden im CD-ROM XA Modus, Form 2 geschrieben. Alle Sessions einer Multisession-CD müssen mit -multi geschrieben worden sein. Eine Session, die man ohne diesen Parameter auf die CD brennt, macht ein späteres Beschreiben unmöglich.

-msinfo
Ermittelt Informationen zur Herstellung von Multisession-CDs und gibt sie aus. Beispiel (mit Output):

[C:\]cdrecord -msinfo dev=4,0
12540,35478

Die erste Zahl steht dabei für den Sektor, an dem eine Session beginnt; die zweite Zahl gibt den ersten beschreibbaren Sektor nach der Session an. Diese beiden Werte werden mkisofs im Parameter -C zur Herstellung eines multisessionfähigen Images übergeben (siehe dort).

-toc
Mit diesem Parameter fragt *cdrecord* die TOC oder PMA einer CD ab und gibt das Ergebnis anschließend aus.

-fix
Bei Angabe dieses Parameters wird die CD nur fixiert. Der Parameter ist sinnvoll, wenn man auf eine CD-R mehrmals geschrieben hat, ohne Sie zu fixieren, um den Abschluß später vorzunehmen.

-nofix
Geben Sie -nofix an, wenn *cdrecord* die CD nach dem Schreibvorgang nicht abschließen soll.

-load
Fährt die Laufwerksschubalde ein. Bei Beginn eines neuen Schreibvorganges wird die Schublade übrigens automatisch eingefahren.

-eject
Wird -eject angegeben, gibt *cdrecord* die CD nach dem Schreibvorgang aus. Wir empfehlen, diesen Parameter immer anzugeben, da einige Recorder erst dann einen neuen Schreibvorgang starten können, wenn die CD nach dem vorherigen Schreibvorgang ausgegeben wurde.

speed=<Wert>
Setzt die Schreibgeschwindigkeit auf <Wert>. 2 steht dabei für Doublespeed, 4 für Quadspeed usw.

blank=<Modus>
Mit Angabe dieses Prameters können Sie eine CD-RW löschen. Wichtige gültige Werte für <Modus> sind:

all - die gesamte CD-RW wird gelöscht
disk - die gesamte CD-RW wird gelöscht
track - nur TOC, PMA und Pregab löschen
session - die letzte Session wird gelöscht
unclose - die letzte Session wieder öffnen
help - zeigt alle Werte für <Modus> an

-force
Sollte eine CD-RW beschädigt sein, ist ein Löschen meist nicht mehr möglich. Mit dem Parameter -force können Sie in den meisten Fällen das Löschen erzwingen.

dev=<device,adapter>
Mit dev wird der Recorder angegeben, der zum Beschreiben der CD-R dienen soll. *device* steht für den SCSI-Adapter im System (0 für den ersten, 1 für den zweiten usw); *adapter* steht für die ID des Recorders. Beispiel:

cdrecord dev=4,0 ...

Damit wird der Recorder mit der SCSI-ID 4 am 1. SCSI-Adapter im System für die folgende Operation benutzt.

timeout=<Wert>
Gibt den Timeout auf dem SCSI-Bus an. Standardwert: 40 Sekunden. Bei oft auftretenden Schreibfehlern kann man <Wert> erhöhen, z.B.:

cdrecord -timeout=80 dev=4,0 ...

-scanbus
Untersucht den SCSI-Bus nach angeschlossenen Geräten und gibt alle gefunde-

nen Geräte mit Ihrer ID aus. Dieser Parameter ist nützlich, wenn man nicht weiß, mit welcher ID der Recorder am Bus angeschlossen ist. Beispiel:

cdrecord -scanbus

-reset
Führt einen Reset des SCSI-Bus durch. Beispiel:

cdrecord -reset

Für abgebrochene Schreibvorgänge, nach denen man die CD-R nicht mehr aus dem Recorder ausgeben kann, oder beim Auftreten anderer Fehler kann dieser Parameter verwendet werden.

Trackparameter

-audio
Ist dieser Parameter angegeben, werden alle im Befehl angegebenen Tracks im Audio-Modus geschrieben. Dateien, die in dieser Weise auf die CD geschrieben werden, sollten mit 16-bit Stereo und 44,1 KHz gesamplet sein. Genaue Informationen zur Herstellung von Audio-CDs mit *cdrecord* finden Sie in Bd. 3 der *OS/2 Only!* im Multimediakapitel.

-data
Alle im Befehl angegebenen Tracks werden im Datenformat auf die CD geschrieben. Dies ist der Standardmodus.

-mode2
Alle Tracks werden im CD-ROM Modus, Form 2 geschrieben.

-xa1
Alle Tracks werden im CD-ROM XA Modus, Form 1 geschrieben.

-xa2
Alle Tracks werden im CD-ROM XA Modus, Form 2 geschrieben.

-cdi
Alle Tracks werden im CDi-Format geschrieben.

Die wichtigsten Parameter von mkisofs
In diesem Abschnitt finden Sie die wichtigsten Parameter von *mkisofs* und ihre Bedeutung:

Parameter zur Spezifikation des Images

-l
Dateinamen dürfen 32 Zeichen lang sein. Dieser Prameter entspricht dem Setzen des ISO-Levels 2

-L
Dateinamen dürfen 32 Zeichen lang sein und mit einem Punkt beginnen.

-J
Erzeugt zusätzlich zum ISO9660-Dateisystem Joliet-Extensions. Kann mit dem Parameter -R kombiniert werden.

-R
Erzeugt zusätzlich zum ISO9660-Dateisystem Rockridge-Extensions. Kann mit dem Parameter -J kombiniert werden. -R ist ein *obligatorischer* Parameter zur Herstellung von Multisession-CDs

-r
Wie Parameter -R, allerdings werden die einzelnen Dateibits auf für Clients sinnvollere Werte gesetzt. Für die Herstellung von CDs unter OS/2 uninteressant. In unserem Falle reicht die Angabe von -R.

-T
Durch Angabe dieses Parameters erzeugt mkisofs eine Datei namens TRANS.TBL in jedem Verzeichnis des CD-Dateisystems, die von Systemen, die standardmäßig keine Unterstützung der Rockridge-Extensions bieten, zur Generierung der korrekten Dateinamen genutzt werden kann. OS/2 macht von derartigen Dateien keinen Gebrauch.

Parameter zur Herstellung des Images

-b <bootimage.img>
Gibt den Namen des Bootimages an, das für die Herstellung eines CD-Images der El

Torito Spezifikation für bootfähige CDs nötig ist. Hinweise zur Praxis in *Bd. 3*.

-x

Mit diesem Parameter kann man Verzeichnisse angeben, die nicht auf die CD übernommen werden sollen. Beispiel:

mkisofs -o image.img -x e:/programme/dos e:/programme

Hier wird das Verzeichnis */dos* im Verzeichnis */programme* nicht mit in das Dateisystem übernommen. Beachten Sie: Dem Parameter -x müssen immer vollständige Pfadangaben übergeben werden, und die auszuschließenden Verzeichnisse müssen sich immer auf das Quellenverzeichnis beziehen. Sie können mkisofs den Parameter -x bis zu 1000 Mal übergeben.

-o

Dieser Parameter gibt die den Zielpfad und die Zieldatei des Images an. Beispiel:

mkisofs -o c:/tmp/image.img e:/programme

In diesem Falle benutzt mkisofs *e:/programme* als Quellverzeichnis und erstellt im Verzeichnis */tmp* auf Laufwerk *c:* das Image mit dem Namen *image.img*.

-v

Gibt während des Betriebes über die Standardausgabe hinausgehende Statusinformationen aus.

-print-size

Ermittelt die Größe des Dateisystems. Benutzen Sie diesen Parameter, wenn Ihr Recorder vor dem Brennvorgang den Umfang des auf die CD zu schreibenden Dateisystems kennen muß, oder wenn Sie sich selbst einen Überblick zu bereits vorhandenen Images verschaffen wollen.

-M <pfad>

Dieser Parameter ist zur Erstellung von Multisession-CDs notwendig. Mit <pfad> wird der Pfad und der Name des Images der vorherigen Session angegeben. Ab der dritten Session gibt man hier den Laufwerksbuchstaben des CD-Recorders an. *mkisofs* benötigt diese Angabe, um die vorherige mit der aktuellen Session zu verbinden.

-C <Sector Start, Sector Ende>

Dieser Parameter ist zur Erstellung von Multisession-CDs *notwendig*. Mit *<Sector Start, Sector Ende>* wird der Output von cdrecord angegeben, wenn das Programm mit dem Parameter -msinfo aufgerufen wurde. *Sektor Start* bezeichnet dabei den Sektor auf der CD, mit dem die vorherige Session beginnt; *Sektor Ende* gibt den ersten beschreibbaren Sektor auf der CD nach der vorherigen Session an. Die Argumente des Parameters sind Dezimalzahlen, z.B.: -C 12450,34257.

Weitere Parameter, die wir aber im folgenden nicht benötigen, finden Sie in den Readme-Dateien der beiden Programme. Für *cdrecord* schreiben wir z.Zt. ein INF-Onlinedokument, das die Anwendung des Programms und aller Parameter erklärt. Dieses Buch wird der Autor seiner Distribution beilegen. Das Buch wird auch eine Referenz zu *mkisofs* beinhalten.

Wie man eine CD-R beschreibt

Nach Vermittlung der nötigen Grundlagen möchten wir nun zur Praxis schreiten und die Herstellung einer *One-Track-CD* betrachten. Alle Daten, die wir auf die CD schreiben, werden dabei in einen Track innerhalb einer Session geschrieben. Solche CDs sind am ehesten unter allen Betriebssystemen austauschbar. Wenn Sie das Image mit ISO-Level 1 herstellen, kann sogar DOS die CD verarbeiten. Für die Goldmasterherstellung können Sie sich genau an die hier angegebene Vorgehensweise halten. Wie anfangs erwähnt, müssen Sie nicht unbedingt ein Image vor dem Brennvorgang herstellen, sondern können beide Vorgänge miteinander kombi-

nieren. Wir beschäftigen uns in einem gesonderten Abschnitt mit diesem Verfahren. Die hier wiedergegebene Vorgehensweise können Sie in ähnlicher Weise auch zur Herstellung von Audio-CDs benutzen. In *Bd. 3* werden wir uns mit der Herstellung eigener Audio-CDs intensiver auseinandersetzen. Um eine normale Daten-CD zu beschreiben, gehen Sie so vor:

❶Erzeugen Sie ein Verzeichnis, z.B.:

```
\datentrack
```

in das sie alle Dateien und Verzeichnisse kopieren, die Sie auf die CD brennen möchten. Wenn Sie einen Goldmaster erstellen möchten, ist diese Vorgehensweise angebracht, da Sie in einem Containerverzeichnis wie *\datentrack* die gesamte Verzeichnisstruktur des Masters zusammenstellen und anschließend, nachdem Sie alle Korrekturen angebracht haben, *mkisofs* übergeben können.

❷Rufen Sie *mkisofs* auf, um ein Image des Verzeichnisses zu erzeugen:

```
mkisofs -o e:/tmp/img.trk -J -R
                                e:/datentrack
```

Damit erstellt mkisofs ein Image, das den Namen *img.trk* trägt und in dem Verzeichnis abgelegt ist, das durch -o spezifiziert ist. Mit -J und -R erhält das Image Joliet- und Rockridge-Extensions. Als Quellverzeichnis dient das am Ende des Befehls angegebene Verzeichnis *e:\datentrack*. Sie müssen natürlich nicht unbedingt ein Containerverzeichnis erstellen, sondern können direkt ein bereits existierendes Verzeichnis angeben, z.B.:

```
mkisofs -o e:/tmp/img.trk -J -R
                                "e:/Meine Daten"
```

❸Legen Sie eine neue CD-R in Ihren Recorder ein und rufen Sie *cdrecord* auf, um das Image auf die CD zu schreiben:

```
cdrecord speed=2 -data -eject -v dev=4,0
                                e:/tmp/img.trk
```

Durch diesen Befehl wird das Image *img.trk* im Verzeichnis *e:\tmp* mit Doublespeed auf die CD-R im Recorder, der mit der SCSI-ID 4 am ersten Adapter im System arbeitet, geschrieben. Der Parameter -data ist eigentlich überflüssig, da er standardmäßig gilt. Wir haben ihn nur der besseren Anschauung wegen angegeben. Die CD wird nach dem Brennvorgang abgeschlossen und anschließend durch den Parameter *-eject* vom Recorder ausgegeben.

Multisession-CDs erstellen

Eine feine Sache wäre die Herstellung von Multisession-CDs, wie man es vom *RSJ CD-Writer* gewohnt ist. Vor allem für Backup-Zwecke oder für die Herstellung eigener Fixpack-CDs, darauf kommen wir später noch, wäre eine Multisession-CD praktisch. Solche CDs kann man auch mit *cdrecord* anfertigen, allerdings ist das nicht ganz so komfortabel. Zu allem Überfluß ist die Dokumentation des Programms zusammen mit der von *mkisofs* ist recht verworren. Außerdem findet man Zusatzinformation und Beispiele zum Brennen von Multisession-CDs nur in einem Dokument, das sich auf Sun-OS und Linux bezieht, nicht aber auf OS/2. Wir brauchten daher recht lange, ehe wir eine Multisession-CD herstellen konnten und benötigten dazu insgesamt 6 Rohlinge. Damit Sie sich dieses Lehrgeld sparen können, finden Sie im folgenden eine Anleitung zum Erzeugen eigener Multisession-

```
cdrecord.exe
 OS/2        Strg+Esc = Fensterliste        HELP = Hilfe
[C:\]cdrecord dev=4,0 speed=2 -multi -eject e:/daten.trk
Cdrecord release 1.8a19 Copyright (C) 1995-1998 J÷rg Schilling
cdrecord: Fifo not supported.
scsidev: '4,0'
scsibus: 0 target: 4 lun: 0
Device type    : Removable CD-ROM
Version        : 2
Response Format: 2
Capabilities   : SYNC
Vendor_info    : 'PHILIPS '
Identifikation : 'CDD3600 CD-R/RW '
Revision       : '2.00'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags   : SWABAUDIO
Starting to write CD/DVD at speed 2 in write mode for multi session.
Last chance to quit, starting real write in 3 seconds._
```

Abb. 11: CDRecord/2 beim Schreiben des ersten Images für eine Multisession-CD. Das Programm wartet vor Beginn des Schreibvorganges 10 Sekunden, um einen Abbruch zu ermöglichen.

CDs. Grundsätzlich beachten Sie zum Brennen solcher CDs bitte die folgenden beiden Punkte:

1. Übergeben Sie *mkisofs* zur Herstellung des Images *immer* den Parameter -R. Weitere Parameter wie -l oder -J können ebenfalls angegeben. *-R ist allerdings obligatorisch. mkisofs* benötigt die Rockridge-Extensions, um die beiden Sessions miteinander zu verbinden.

2. Stellen Sie sicher, daß Sie alle Tracks als XA-Tracks auf die CD schreiben. Sie erreichen dies durch Angabe des Parameters *-multi*. *cdrecord* setzt dann automatisch den richtigen Trackmodus. Die Angabe der Parameter -xa1 oder -mode2 können Sie sich also sparen. Den Parameter -xa2 anzugeben, empfehlen wir nicht. Einmal ist es unnötig, weil der schon in *-multi* enthalten ist; zum anderen bricht das Programm mit Fehlern ab, wenn der Schreibvorgang beginnt.

In unserem Beispiel soll eine CD mit nur zwei Sessions erstellt werden. Später lassen sich noch weitere Sessions hinzufügen. Wir raten generell jedoch nicht dazu, mehr als 10 Sessions auf die CD zu brennen. Das ist keine Einschränkung von cdrecord, sondern ein Hinweis, der generell beachtet werden sollte. Zur Herstellung einer Multisession-CD verfahren Sie wie folgt:

➊ Erstellen Sie zwei Verzeichnisse, welche die Daten der einzelnen Sessions aufnehmen, z.B.:

\session1
\session2

➋ In diese Verzeichnisse kopieren Sie die Daten, die auf die CD gebrannt werden sollen, z.B. erzeugen Sie folgende Verzeichnisse:

\session1\Daten
\session2\Noch mehr Daten

Anschließend füllen Sie die beiden Unterverzeichnisse von *\session1* und *\session2*, also *\Daten* und *\Noch*

mehr Daten mit einzelnen Dateien. Wir haben zum Test die Dateien aus dem \OS2- und \OS2\DLL-Verzeichnis in die beiden Containerverzeichnisse kopiert. Selbstverständlich können in den vorbereiteten Verzeichnissen auch komplexere Verzeichnisstrukturen angelegt werden.

❸Rufen Sie anschließend *mkisofs* auf, um das Image der ersten Session zu erzeugen:

```
mkisofs -o e:\tmp\img1.trk -J -R
                    e:\session1
```

Damit wird im Verzeichnis *\tmp* auf Laufwerk *e:* die Imagedatei *img1.trk* erzeugt, welche die Daten aus dem Verzeichnis *\session1* auf dem Laufwerk *e:* enthält.

❹Schreiben Sie die erste Session mit cdrecord wie folgt auf die CD:

```
cdrecord speed=2 -multi -v -eject
                dev=4,0 e:\tmp\img1.trk
```

Dann schreibt *cdrecord* auf den Recorder mit der SCSI-ID 4 am 1. Adapter im System mit Doublespeed die Daten des Images *img1.trk* im Multisessionmodus auf die CD-R und gibt den Datenträger nach dem Abschließen der Session aus. Den Beginn des Schreibvorganges dokumentiert *Abb. 11*.

❺Rufen Sie *cdrecord* noch einmal auf, um die Sektorinformationen der ersten Session für *mkisofs* zu erhalten:

```
cdrecord -msinfo dev=4,0
```

Die Multisessioninformationen werden dann ausgegeben, etwa als:

```
0,12409
```

Dabei steht die erste Ziffer für die Nummer des Sektors, mit dem die Session beginnt; die zweite Ziffer gibt die Nummer des Sektors an, mit dem die Session endet und bezeichnet die Stelle, ab der *cdrecord* das Image der nächsten Session auf die CD-R schreiben kann.

❻Erzeugen Sie mit *mkisofs* und den zuvor mit *cdrecord* abgefragten Sessioninformationen das Image für die 2. Session, die auf der CD erscheinen soll:

```
OS/2-Fenster
 OS/2          Strg+Esc = Fensterliste          HELP = Hilfe
[C:\]cdrecord dev=4,0 -msinfo
22173,32964

[C:\]mkisofs -o e:\daten.trk -l -L -R -J -C 22173,32964 -M f: e:/daten/bücher
Total extents scheduled to be written = 43837
 45.58% done, estimate finish Sun Apr 25 20:17:21 1999
 52.09% done, estimate finish Sun Apr 25 20:17:41 1999
 58.60% done, estimate finish Sun Apr 25 20:17:52 1999
 65.12% done, estimate finish Sun Apr 25 20:18:03 1999
 71.62% done, estimate finish Sun Apr 25 20:18:11 1999
 78.13% done, estimate finish Sun Apr 25 20:18:18 1999
 84.65% done, estimate finish Sun Apr 25 20:18:23 1999
 91.14% done, estimate finish Sun Apr 25 20:18:27 1999
 97.67% done, estimate finish Sun Apr 25 20:18:32 1999
Total extents actually written = 43837
Total translation table size: 0
Total rockridge attributes bytes: 36895
Total directory bytes: 88064
Path table size(bytes): 414
76801 extents written (150 Mb)

[C:\]_
```

Abb. 12: Abfragen der Sessioninformationen mit cdrecord und Erstellen des Images für die 2. Session mit mkisofs.

```
mkisofs -o e:\tmp\img2.trk -J -R
     -C 0,12409 -M f: e:\session2
```

Damit erzeugt *mkisofs* im Verzeichnis *\tmp* auf Laufwerk e: die Imagedatei namens *img2.trk*. Das Image wird dabei mit dem der 1. Session verbunden. Dazu dienen die Parameter -C und -M, die immer zusammen angegeben werden. -C gibt die Sessioninformationen an, die Sie mit *cdrecord* abgefragt haben. -M spezifiziert den Laufwerksbuchstaben des CD-Recorders, in unserem Falle *f:*, wobei der Doppelpunkt angegeben werden muß, aber keine Pfadangabe. Anstelle des Laufwerksbuchstabens kann man auch den Namen des ersten Images angeben, muß jedoch mindestens einen Schrägstrich (Forwardslash) / voranstellen, z.B. /img1.trk, damit *mkisofs* auf die Daten der ersten Session zugreifen kann.

! ***Geben Sie am besten stets direkt den Laufwerksbuchstaben an, da man in diesem Falle die erste Imagedatei nach dem Brennen löschen kann. Außerdem müssen Sie bereits ab der 3. Session den Laufwerksbuchstaben des Recorders angeben.***

Den Vorgang des Abfragens der Sessioninformationen und des Erzeugens des zweiten Images zeigt *Abb. 12*.

❼ Schreiben Sie nach Abschluß der 1. Session die zweite mit *cdrecord* wie folgt auf die CD:

```
cdrecord speed=2 -multi -v -eject
     dev=4,0 e:\tmp\img2.trk
```

Nach dem Abschließen der 2. Session wechseln Sie auf das CD-Recorder Laufwerk und geben *dir* ein. Sie werden dann zwei Verzeichnisse auf der CD vorfinden, in diesem Falle *\Daten* und *\Noch mehr Daten*.

❽ Verfahren Sie mit weiteren Sessions genau so. Den Schreibvorgang der 2. Session zeigt *Abb. 13*.

Wie Sie sehen, gibt sowohl *mkisofs* als auch *cdrecord* genaue Statusinformationen zu den gerade durchgeführten Aktivitäten aus. Über das, was gerade geschieht, ist man also besser im Bilde als mit dem *RSJ CD-Writer*.

Natürlich ist diese Vorgehensweise nicht sehr komfortabel, v.a. für all jene nicht, die mit der Kommandozeile auf Kriegsfuß stehen; aber man kann den Brennprozeß auch mit einem kleinen REXX-Programm oder einer Batchdatei weitgehend automatisieren. Das ist relativ einfach. Aus Platzgründen können wir jedoch nicht mehr in diesem Band auf diesen interessanten Aspekt eingehen und belassen es bei den Schritt-für-Schritt-Anleitungen für die wichtigsten Operationen mit den beiden Programmen.

Wir möchten zu den Multisession-CDs noch eine interessante Anmerkung machen: Sollten Sie nach dem Brennvorgang nur die erste Session erkennen können, nehmen Sie die CD aus dem Recorder und versuchen Sie, den Datenträger mit einem normalen CD-ROM-Laufwerk zu lesen. *cdrecord* schreibt die zweite Session korrekt auf die CD. Manche Recorder können die zweite und alle folgenden Sessions nur nicht lesen, auch wenn sich das vielleicht merkwürdig anhört. Eines dieser Geräte ist z.B. unser *Philips CDD3600* oder mit ihm verwandte Geräte. Der Grund hierfür liegt in Unstimmigkeiten zwischen dem Einheitenmanager OS2CDROM.DMD und dem Laufwerk. Die Multisession-CDs sind aber völlig

korrekt beschrieben. Wir haben Sie mit verschiedenen SCSI-CD-ROM-Laufwerken (SCSI-Modelle von TEAC, NEC) und einem ATAPI-Laufwerk von Toschiba getestet. Alle Sessions konnten erkannt werden. Nur war unsere Verwunderung groß, daß gerade der Recorder nicht mit der CD umgehen konnte.

In aller Regel sind die CDs also mit herkömmlichen CD-ROM-Laufwerken lesbar. Wir testeten es weiter aus, indem wir auch mehr als nur zwei Sessions auf die CD brannten und den Recorder wechselten. Mit dem bereits vorgestellten Modell von *Yamaha* trat gegenüber dem *Philips*-Recorder z.B. nicht das Problem auf, nur die erste Session der CD lesen zu können. Hier erkannte auch der Recorder alle Sessions. Ferner sei noch angemerkt, daß diese Fehler mit ATAPI-CD-ROM-Laufwerken nicht auftreten dürften.

Ob die Multisession-CD erkannt wird, ist also betriebssystem- und laufwerkssbhängig und hier nicht nur vom Modell sondern auch vom Typ (SCSI oder IDE). Das hat aber weniger mit *cdrecord* oder *mkisofs* zu tun, sondern ist die allgemeine crux der Multisessiontechnologie.

```
cdrecord.exe
scsidev: '4,0'
scsibus: 0 target: 4 lun: 0
atapi: 0
Device type    : Removable CD-ROM
Version        : 2
Response Format: 2
Capabilities   : SYNC
Vendor_info    : 'PHILIPS '
Identifikation : 'CDD3600 CD-R/RW '
Revision       : '2.00'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags   : SWABAUDIO
Drive buf size : 786432 = 768 KB
FIFO size      : 4194304 = 4096 KB
Track 01: data    7 MB
Total size:       8 MB (00:51.05) = 3829 sectors
Lout start:       8 MB (00:53/04) = 3829 sectors
Current Secsize: 2048
  ATIP start of lead out: 336075 (74:43/00)
Blocks total: 336075 Blocks current: 252372 Blocks remaining: 248543
Starting to write CD/DVD at speed 2 in write mode for multi session.
Last chance to quit, starting real write in 1 seconds.
Starting new track at sector: 83703
Track 04:   6 of    7 MB written.
```

Abb. 13: CDRecord/2 beim Schreiben weiterer Sessions. Das Programm gibt eine Fülle von Statusinformationen aus.

CDs ohne Image brennen

Wie erwähnt müssen Sie nicht unbedingt ein Image erstellen, und anschließend *cdrecord* aufrufen, um es auf die CD zu brennen. Wir empfehlen Ihnen, so zu verfahren. Sie wissen so genau, wieviel Speicherplatz auf dem Medium noch zur Verfügung steht. Trotz allem muß man diesen Schritt nicht tun.

Das Programm *mkisofs* kann das Image auch in eine Pipe statt eine Datei schreiben, die *cdrecord* ausliest, um den Inhalt auf die CD zu brennen. Das bedeutet aber:

- Das System muß in der Lage sein, den Datenstrom aufrecht zu erhalten, d.h. es muß die Pipe immer füllen können. Aus der Sicht der Applikation ist eine Pipe ein Ringpuffer, in den eine Serveranwendung schreibt, während die Clientapplikation daraus liest. Das Schreiben des Servers und das Lesen des Clients laufen dabei gleichzeitig ab. Ist der Puffer leer, weil die Systemleistung nicht ausreicht, ihn zu füllen, kann der Client nicht mehr aus ihm lesen. Die Folge ist dann ein Timeout, d.h. der Schreibprozeß wird unterbrochen. Passiert so etwas während

des Schreibens auf die CD, wird der Datenträger unbrauchbar. Man kann in einem solchen Fall versuchen, den Puffer, den *cdrecord* zum Schreiben verwendet, zu vergößern.

- Der Prozeß braucht mehr Rechenzeit, seine Priorität muß daher erhöht werden.
- Zur Vermeidung von Timeouts sollte ein Puffer angegeben werden, der um so größer sein sollte, je kleiner die Systemleistung ist.
- Während des Schreibvorganges sollten leistungsschwächere Systeme nicht übermäßig bzw. gar nicht beansprucht werden.

Wenn Sie diese Richtlinien beachten, steht einem »on-the-fly«-Brennvorgang mit *cdrecord* nichts mehr im Wege. Auf jeden Fall benötigt der Schreibprozeß eine höhrere Priorität, diese Bedingung findet man auch in den README-Dateien. Wir haben uns davon nicht abschrecken lassen, und den Brennvorgang ohne eine Erhöhung der Priorität gestartet - ohne Erfolg.

Standardmäßig existiert unter OS/2 kein Befehl, mit dessen Hilfe Sie die Priorität eines Prozesses verändern können. Aber wie für nahezu jeden Zweck gibt es ein Freewareprogramm, das dieses Problem beseitigt. Es heißt *spe*. Wir haben es ebenfalls auf unserer Homepage für Sie bereitgelegt, Sie finden es aber auch in den bekannten Softwarearchiven LEO oder Hobbes.

Zur Installation entpacken Sie das Zip-Archiv und kopieren das Programm am besten in der Verzeichnis \OS2 Ihres Bootlaufwerkes. Anschließend können Sie *mkisofs* und *cdrecord* zusammen mit *spe* aufrufen. Geben Sie auf der Kommandozeile folgenden Befehl ein, um ohne die Herstellung eines Images Daten auf die CD zu schreiben:

```
spe t28 mkisofs -R -J e:\daten |
    cdrecord speed=2 -data fs=8m dev=4,0
```

Mit *spe t28* erhöhen Sie die Priorität des Prozesses, in dem *mkisofs* läuft, auf das Level 28. Sie können die Priorität auch maximal auf Level 30 erhöhen, jedoch nicht mehr. Da *cdrecord* in der hier verwendeten Version auf Level 31 läuft, würde die gleiche oder eine höhere Priorität zu einem Überlauf der Pipe führen, und *cdrecord* kommt mit dem Schreiben nicht mehr hinterher. Die Parameter -l, -L, -R und -J von *mkisofs* kennen Sie bereits, so daß wir nicht näher darauf eingehen müssen. Die Angabe *e:\daten* gibt in unserem Beispiel das Laufwerk und das Verzeichnis an, in dem sich die Quelldateien befinden. Das Symbol | definiert die Pipe. Sie kennen es vielleicht in Verbindung mit dem Befehl *more*, z.B.:

```
type readme.txt | more
```

Hier schreibt der Befehl *type* die Daten aus der Datei *readme.txt* nicht auf den Bildschirm sondern in eine Pipe, auf die das Programm *more* zugreift, um eine seitenweise Ausgabe des Dateiinhaltes auf dem Bildschirm zu ermöglichen. In unserem Beispiel schreibt *mkisofs* in eine Pipe, und *cdrecord* liest Sie aus. Die anderen Parameter kennen Sie bereits.

An dieser Stelle seien ein paar Worte zum Puffer von *cdrecord* erwähnt, der mit dem Parameter fs gesetzt wird. Standardmäßig ist dieser Puffer 4 MByte groß. Dies reichte auf unserem 486er-Testsystem bereits aus. Wenn Sie die Größe des Puffers verändern möchten, benutzen Sie den Parameter fs wie folgt:

fs=<Wert> [b, k, m, s, f]

<*Wert*> ist dabei die Größe des Puffers in Byte. Folgt <*Wert*> ein *b* wird <*Wert*> mit 512, folgt ihm ein *k* mit 1024, folgt ihm ein *m* mit 1024^2, folgt ihm ein *s* mit 2048 und folgt ihm ein *f* mit 2352 multipliziert. Der im Beispiel angegebene Puffer ist demnach 8 MByte groß. Sie sollten im Normalfall mit der Standardgröße des Puffers auskommen. Es sei Ihnen ferner empfohlen, den Puffer nicht größer als 8 MByte anzusetzen.

! ***Beachten Sie den Bindestrich am Ende des Befehls. Er ist notwendig, sonst arbeitet* cdrecord *nicht.***

Nach Übergabe dieses Befehls an den Interpreter starten die Programme. *Mkisofs* überprüft zunächst die Dateierweiterungen und bereitet die Fertigung des Images vor, das in die Pipe geschrieben wird. Beim Start von *cdrecord* wird ein Warnhinweis ausgegeben, der Sie davon unterrichtet, daß die Daten evtl. nicht auf die CD passen, da *cdrecord* logischerweise die Größe des Images nicht kennt (da es ja noch gar nicht existiert).

! ***Vergewissern Sie sich vor dem Brennen, daß auf der CD genügend Platz frei ist, um die Daten aufzunehmen.***

Dann wird der Schreibprozeß wie gewohnt gestartet. Beachten Sie bitte die oben angegebenen Hinweise, wenn Sie mit dem System weiterarbeiten möchten. Wir fanden auf eiener Webpage den Hinweis, daß ein Beschreiben von CDs »on-the-fly« mit Multitasking-Systemen wie OS/2 nicht möglich sei, selbst wenn ein schnelles Festplattensystem zum Einsatz käme. *Diese Aussage ist definitiv falsch*. Bereits auf einem 486er mit ausreichendem Festspeicherausbau können Sie CDs nach der geschilderten Methode brennen und nebenbei normal mit dem System arbeiten.

In diesem Zusammenhang sei natürlich noch erwähnt, daß Sie auch Multisession-CDs auf diese Art und Weise anfertigen können. Damit ersparen Sie sich auch hier den Umweg über die Herstellung eines Images.

Mit einer Pipe zu arbeiten, ist bequemer als mit Images und eignet sich besonders gut zur Automatisierung von Backup-Prozessen. Zur besseren Handhabung sollte man sich mit Hilfe einiger Batchdateien je nach den eigenen Bedürfnissen Backup-Kataloge bereitstellen und die beiden Programme mit dem angegebenen Befehlskonstrukt während der Stapelverarbeitung aufrufen. Wer gerne programmiert, kann sich seine eigene Backup-Lösung mit den beiden Programmen rasch selbst zusammenstellen.

Vorteile gegenüber dem RSJ CD-Writer

Im Gegensatz zum *RSJ CD-Writer* lassen sich mit dem *CDRecord/2* viel problemloser CDs erstellen, bei denen sich sämtliche Daten in nur einem Track befinden. Das ist immer dann von Vorteil, wenn man einen hochwertigen Goldmaster für die Serienproduktion herstellen möchte. Fertigt man eine Daten-CD mit dem *RSJ CD-Writer*, findet man auf der beschriebenen CD-R stets ca. 11 bis 18 Tracks. Das läßt sich nicht vermeiden, es sei denn man erstellt erst mit *mkisofs* ein Image, das man dann mit *CDView* in einem Zuge auf die CD schreiben kann (s. den Abschnitt zu *CDView*). *RSJ* empfiehlt, die Daten von einer mit dem *CD-Writer* hergestellten CD vor dem Glasmastering in einen Track zu kopieren. Verwenden Sie *cdrecord* und mkisofs, können Sie sich dieses Procedere sparen.

Ein weiterer Vorteil gegenüber dem Programm von *RSJ*: Der CD-Recorder steht als ganz normales Laufwerk jeder-

zeit zur Verfügung. Man kann eine CD einlegen und von ihr lesen, um dann später damit aufzuzeichnen. Kunstgriffe, wie sie beim *RSJ CD-Writer* vonnöten sind, entfallen. Kann man sich mit der Kommandozeile anfreunden, ist ein bedeutend flexibleres Arbeiten mit *CDRecord/2* möglich.

Ebenfalls praktisch ist die Ausgabe von Statusinformationen. Man weiß genau, wieviel MByte bereits auf die CD geschrieben wurden. Zu jedem Prozeß wird die benötigte Zeit mit ausgegeben, zum Beschreiben und auch zum Abschluß einer Session. Da *CDRecord/2* auch Multisession-CDs herstellen kann, dürfte er dem *RSJ CD-Writer* in nichts nachstehen. Auch die Fähigkeit, Audio-CDs recht einfach zu erstellen, macht *CDRecord/2* sehr interessant. Übrigens unterstützt *CDRecord/2* wie der *RSJ CD-Writer* viele CD-Recorder (auch die von uns getesteten). Allerdings werden ATAPI-Laufwerke nicht unterstützt. Das ist jedoch nicht weiter bedauerlich, da man ohnehin zu einem SCSI-Recorder greifen sollte.

Aber kein Licht ohne Schatten. Ein Dateisystem wie der *RSJ CD-Writer* für den Zweck des CD-Recording zu verwenden hat auch Vorteile. Etwa die fehlende OS/2-Unterstützung für die Rockridge Extensions. Betrachtet man sich die erstellten CDs mit dem *RSJ CD-Writer*-Dateisystem kann man diesen Nachteil umgehen. Dennoch bleibt das immer noch recht umständliche An- und Abmelden des Recorders mal als CD-ROM- Laufwerk und mal als Gerät zur Aufzeichnung.

Nicht ein wirklicher Schwachpunkt, aber für den Minimalanwender u.U. wichtig, ist die fehlende Unterstützung von ATAPI-Geräten. Hat man einen solchen CD-Recorder, kann man *cdrecord* nicht verwenden. Wer jedoch hohe Funktionalität verbunden mit Flexibilität benötigt, der kommt um dieses Programm kaum herum. Auf jeden Fall hat uns *cdrecord* und *mkisofs* so gut gefallen, daß wir für beide Programme eine graphische Benutzeroberfläche erstellen.

Wie der *RSJ CD-Writer* werden auch *mkisofs* und *cdrecord* beständig weiterentwickelt. Der Autor der OS/2-Version von *mkisofs*, Robert Lalla, hat z.B. angekündigt, eine echte Unterstützung von Sonderzeichen für die Joliet-Extensions zu entwickeln. Bisher ist die Umsetzung von der OS/2 Codepage (850) auf Unicode nur für den Standard-ASCII-Zeichensatz korrekt, nicht jedoch für Umlaute oder andere Sonderzeichen. Hilfe bekommt man von den Autoren übrigens auch, wenn man überhaupt nicht mehr weiterkommen sollte, weil etwa ein Bug die Arbeit behindert.

Abschließende Betrachtungen

Welches Programm sollte man nun verwenden? Das ist eine Frage, die nur sehr schwierig zu beantworten ist. Der *RSJ CD-Writer* hat seine Vorteile hinsichtlich des Bedienkomforts, allerdings scheitert das Dateisystem an der einfachen Aufgabe, eine One-Track-CD zu brennen. *CDRecord/2* hat damit zwar keine Probleme; dafür braucht man für die Einarbeitung in das Programm schon etwas mehr Zeit. Auf der anderen Seite läßt es sich aber auch in eigene Programme einbauen, und es existieren auch graphische Schnittstellen, die wir Ihnen in *Bd. 3* im Multimediakapitel ausführlich vorstellen. Audio- wie Daten-CDs lassen sich mit beiden Programmen in hoher Qualität erzeugen. Mit der Joliet-Unterstützung von *mkisofs* hapert es noch ein wenig, aber der Autor ist dabei, eine korrigierte Unterstützung bereitszustellen. Das Fertigen von Multisession-CDs ist weder für den *RSJ-Writer* noch für *cdrecord* ein Problem.

Da man mit *cdrecord* gegenüber dem *RSJ CD-Writer* also nichts verliert, und beim *RSJ-Produkt* in manchen Fällen (One-Track-Disks) ohnehin auf *mkisofs* zurückgreifen muß, da man mit dem Dateisystem leider nicht alles machen kann, wird es letztlich wohl eine Frage des Geldbeutels sein, für welche der Lösungen man sich entscheidet. Während *cdrecord* und *mkisofs* ausgezeichnete Freeware ist, stellt der *RSJ CD-Writer* schon eine recht hohe Ausgabe dar, die zwar ihr Geld wert ist - aber wenn man nur ein schmales Budget zur Verfügung hat und sicher schon einiges in einen CD-Recorder investierte, so sollte man ruhig zu *cdrecord* greifen. Man hat damit die gleichen Möglichkeiten wie mit dem kommerziellen Produkt. Gewisse Pferdefüße beider Produkte stehen sich dabei gegenüber.

Informationen zu mkisofs

Beziehbar in der jeweils aktuellen Version über alle bekannten OS/2-Softwarearchive, wie LEO oder Hobbes, oder unsere Homepage.

Kontakt mit dem Autor:
rlalla@stepnet.de

Informationen zu CDRecord/2

Beziehbar in der jeweils aktuellen Version über alle bekannten OS/2-Softwarearchive, wie LEO oder Hobbes, oder unsere Homepage.

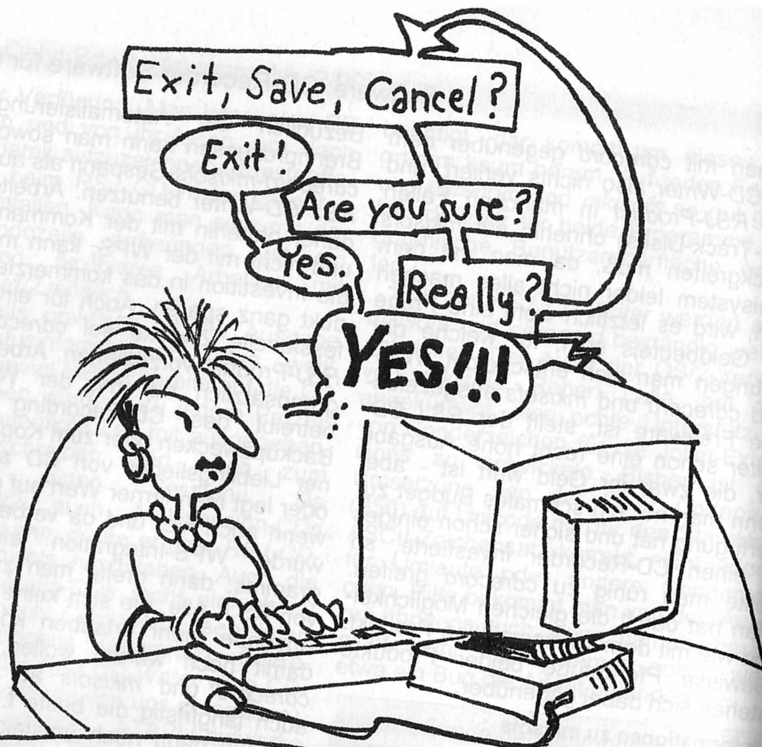
Homepage der Original Unix-Version:
http://www.fokus.gmd.de/nthp/employees/schilling/cdrecord.html

Homepage von CDRecord/2:
http://www.geocities.com/SiliconValley/Sector/5785/

Kontakt mit dem Autor:
christopher.wohlgemuth@bch.siemens.de

Bezüglich der Automatisierung von Brennprozessen kann man sowohl das *cdrecord-mkisofs*-Gespann als auch den *RSJ CD-Writer* benutzen. Arbeitet man daher ohnehin mit der Kommandozeile und nicht mit der WPS, kann man sich die Investition in das kommerzielle Produkt ganz sparen. Auch für einige professionelle Zwecke sei *cdrecord* dem *RSJ*-Produkt vorzuziehen. Arbeitet man grundsätzlich nur mit der WPS und betreibt das CD-Recording nur zu Backupzwecken oder zum Kopieren seiner Lieblingslieder von CD auf CD-R, oder legt man ferner Wert auf eine gute, wenn auch hier und da verbesserungswürde WPS-Integration eines Programms, dann greife man zur Lösung von RSJ. Alle, die sich keine finanzielle Ausgabe mehr erlauben können oder damit noch warten wollen, finden in *cdrecord* und *mkisofs* auf jeden Fall auch langfristig die beste Lösung (und ob man dann noch umsteigen will...?). Allen Programmierern sei nochmals empfohlen, sich *cdrecord* und *mkisofs* einmal näher anzusehen. Die Einbindung der beiden Produkte in eigene Programmoberflächen ist für jeden Einsatzzweck denkbar einfach und für jeden Entwickler interessant. Da der *RSJ CD-Writer* auch weit verbreitet ist, sei den Entwicklern auch empfohlen, neue graphische Frontends zu schreiben, die mit dem Dateisystem auf noch bessere Weise zusammenarbeiten als die Lösungen von *RSJ*.

Eine Entscheidung zwischen beiden Programmen zu treffen, sei also letztlich nach einem persönlichen Test jedem selbst überlassen. Auf jeden Fall lohnen beide CD-Recording-Lösungen eine nähere Betrachtung. Und: Es ist sehr angenehm zu wissen, daß es gerade für den Anwendungsbereich des CD-Recordings für Warp so gute, OS/2-konforme Software gibt. ❑


Exit, Save, Cancel?
Exit!
Are you sure?
Yes.
Really?
YES!!!

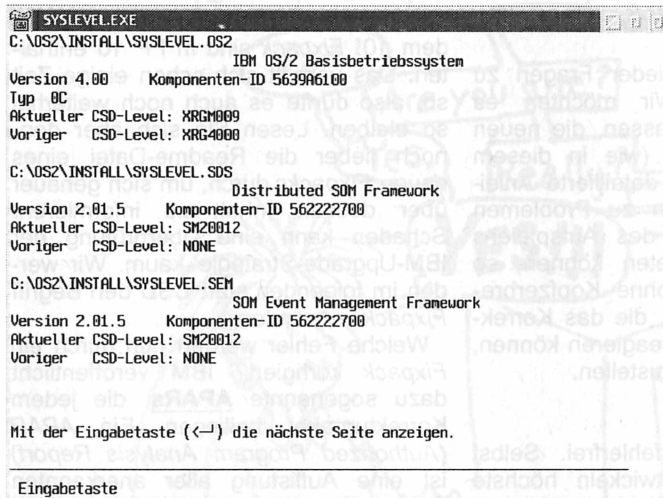
Ich HASSE
diese ADAPTIVEN PROGRAMME,
die immer besser wissen als
ich selber, was gut für mich ist!

Alles über Fixpacks

Uns haben immer wieder Fragen zu *Fixpacks* erreicht. Wir möchten es daher nicht dabei belassen, die neuen Fixpacks vorzustellen (wie in diesem Band), sondern auch detaillierte Anleitungen und Lösungen zu Problemen bieten, die während des Aufspielens eines Fixpacks auftreten können, so daß Sie in Zukunft ohne Kopfzerbrechen auch auf Fehler, die das Korrekturprogramm meldet, reagieren können, um deren Ursache abzustellen.

Einige Worte zuvor
Kein Programm ist fehlerfrei. Selbst wenn man beim Entwickeln höchste Sorgfalt walten läßt - Fehler schleichen sich doch immer wieder in den Programmcode ein. Um sie zu beseitigen, nimmt man ein Update des eingesetzten Programms vor, das die Fehler der Vorgängerversion korrigiert (manchmal ist das leider nicht immer der Fall). Was OS/2 betrifft, so geschieht dies mit **CSDs** *(Corrective Service Disks)* oder *Fixpacks*. Genau genommen sollte man zwischen *CSDs* und *Fixpacks* unterscheiden. Ein *CSD* erhöht den Service-Level oder sogar die Versionsnummer des Betriebssystems; ein *Fixpack* eigentlich nicht. *CSDs* sind kumulativ, d.h. das aktuelle *CSD* beinhaltet alle Fixes der vorangegangenen *CSDs*; *Fixpacks* sind nicht kumulativ. Allerdings sind die Übergänge zwischen den Begriffen mittlerweile völlig fließend: IBM bezeichnet die freigegebenen Korrekturen als *Fixpacks*, aber der *Syslevel* wird erhöht, womit es eigentlich *CSDs* sind. Daneben sind *Fixpacks* kumulativ. Durch die etwas merkwürdige Terminologie braucht man sich also nicht verunsichern zu lassen: Wenn Sie das *Fixpack 10* für Warp 4 aufspielen, wird der Service-Level Ihres Systems entsprechend erhöht, und alle Korrekturen vor dem 10. *Fixpack* sind in FP 10 enthalten. Das verhält sich schon einige Zeit so, also dürfte es auch noch weiterhin so bleiben. Lesen Sie sich aber dennoch lieber die Readme-Datei eines neuen *Fixpacks* durch, um sich genauer über dessen Inhalt zu informieren. Schaden kann eine Überprüfung der IBM-Upgrade-Strategie kaum. Wir werden im folgenden statt *CSD* den Begriff *Fixpack* verwenden.

Welche Fehler werden nun durch ein *Fixpack* korrigiert? IBM veröffentlicht dazu sogenannte **APARs**, die jedem Korrekturpaket beiliegen. Ein *APAR (Authorized Program Analysis Report)* ist eine Auflistung aller anerkannten Fehler eines Produktes. *APARs* sind im übrigen ebenfalls kumulativ, d.h. sie enthalten die Beschreibungen von Bugs vom ersten bis zum aktuellsten *Fixpack*. Wenn man sehen möchte, ob ein bestimmter Fehler in den *APARs* steht, beginnt eine recht mühsame Suche durch die umfangreiche Datei. Allerdings schadet es nicht, wenn man den *APAR* durcharbeitet, um zu sehen, ob ein *Fixpack* überhaupt aufgespielt werden muß. Meistens ist das nämlich nicht der Fall, es sei denn, es enthält neue Treiber oder Dienstprogramme oder führt grundlegende Änderungen am Betriebssystem durch, die ein sichereres Arbeiten mit dem System ermöglichen. Solche Informationen findet man in der Readme-Datei zum *Fixpack*, so daß man auch nach der Lektüre dieser Datei entscheiden kann, ob die Installation des *Fixpacks* nötig ist oder nicht. Was die aktuellen *Fixpacks* angeht: Hier sollte jeder die Installation durchführen, da die Änderungen am System schon beachtlich sind, wie wir im Kapitel *Neuigkeiten* ausgeführt haben. Es kann also durchaus sein, daß durch ein *Fixpack* OS/2 mit gänzlich neuen Funktionen



```
SYSLEVEL.EXE
C:\OS2\INSTALL\SYSLEVEL.OS2
                              IBM OS/2 Basisbetriebssystem
Version 4.00     Komponenten-ID 5639A6100
Typ  0C
Aktueller CSD-Level: XRGM009
Voriger   CSD-Level: XRG4000

C:\OS2\INSTALL\SYSLEVEL.SDS
                              Distributed SOM Framework
Version 2.01.5     Komponenten-ID 562222700
Aktueller CSD-Level: SM20012
Voriger   CSD-Level: NONE

C:\OS2\INSTALL\SYSLEVEL.SEM
                              SOM Event Management Framework
Version 2.01.5     Komponenten-ID 562222700
Aktueller CSD-Level: SM20012
Voriger   CSD-Level: NONE

Mit der Eingabetaste (<─┘) die nächste Seite anzeigen.

Eingabetaste
```

Abb. 1: Eine Bildschirmseite von *syslevel.exe* Diese Seite zeigt den Service-Level eines Warp 4-Systems, auf das FP 9 aufgespielt wurde. Die Informationen stammen aus der Datei SYSLEVEL.OS2 im Verzeichnis \os2\install.

bereichert wird, auch wenn dies nicht die Philosophie von IBM ist. Es wäre eine Ausnahme der Regel, wie sie auch in den aktuellen *Fixpacks* mit den neuen REXX-Funktionen auftritt. Gerüchte um einen neuen OS/2-Client gibt es nahezu überall in den unterschiedlichsten Ausführungen. Jedoch gibt es seitens IBM definitiv keine Aussage zu diesem Thema. Es könnte jedoch rein technisch möglich sein, ein Update z.B. auf ein Warp 4.5 mit einem Fixpack durchzuführen, ähnlich wie das *CSD* für OS/2 2.1 die Versionsnummer auf 2.11 erhöhte. Aber wir beschreiben hier nur Möglichkeiten, keine Fakten zum weiteren Zukunft des Warp 4 Client.

Auf welchem Level ist das System?

Jeder kennt sicherlich das Programm *syslevel*. Mit ihm kann man den Service-Level aller auf dem OS/2-System installierten Produkte abfragen, die eine *syslevel*-Datei haben. Das sind neben dem Basisbetriebssystem, dem Graphiksubsystem und dem MMPM auch der Netzwerk- und Java-Support. Aber auch andere IBM-Produkte werden durch *syslevel* angezeigt, etwa der Service-Level des *IBM C-Set*. Um die Service-Level der einzelnen Produkte zu erhalten, öffnet man ein OS/2-Fenster und gibt dort den Befehl *syslevel* ein. Dann durchsucht das Programm alle Festplatten nach den *syslevel*-Dateien, also auch die Betriebssystempartitionen, die gerade nicht aktiv sind (wenn Sie z.B. unterschiedliche OS/2-Versionen auf mehreren Laufwerken eines Systems installiert haben). Je nach Anzahl der Partitionen und deren Größe kann *syslevel* recht lange brauchen, ehe es die entsprechenden Dateien gefunden und die darin enthaltenen Informationen angezeigt hat. Die Anzeige erfolgt Bildschirmweise wie in *Abb. 1*. Aus der Vielzahl der angezeigten Informationen interessieren uns nur der *CSD-Level* des Basisbetriebssystems und dessen *Service-Level*. In unserem Beispiel haben wir *syslevel* auf einem Warp 4-System ausgeführt, auf dem zuvor ein *FP 9* installiert wurde. Wie man sieht, beinhaltet die Datei *syslevel.os2* alle wichtigen Angaben zum Basisbetriebssystem: Die Versionsnummer und auch den aktuellen wie den vorherigen *CSD-Level*. Der vorherige Betriebssystemlevel war *XRG4004*, was ein deutsches Warp 4 bezeichnet. Der aktuelle *CSD*-

Level ist *XRGM009*, was auf die Installation unseres Fixpacks hinweist. Woher weiß man aber, daß es sich um ein *FP 9* handelt? Einfach: Die Datei *syslevel.fpk* enthält Informationen über den aufgespielten *Service-Level*. *Abb. 2* zeigt die Informationen, die *syslevel.exe* aus dieser Datei gelesen hat. Man kann hieraus ersehen, daß ein Fixpack mit dem *CSD-Level XRGM009* aufgespielt wurde. Vorheriger und aktueller *CSD-Level* sind in diesem Falle gleich. Im wesentlichen sind es diese beiden Dateien, die auch für das Korrekturprogramm von Interesse sind.

Es sollte vielleicht noch erwähnt werden, daß es unter Warp 4 keine *syslevel*-Datei für den MMPM mehr gibt, da die Multimediaunterstützung ab Warp 4 fester Bestandteil des Betriebssystems ist. Unter Warp 3 enthält die Datei *syslevel.mpm* den *CSD-Level* des MMPM. Wundern Sie sich also nicht, falls Sie während der folgenden Fehlerkorrekturen Ihres Warp 4-Systems den MMPM nicht mehr finden. Möchten Sie ein Warp 3-System korrigieren, lesen Sie unbedingt die Datei Readme durch, die Sie im *Fixpack* finden. Unter Warp 3 wird der MMPM als eigenständiges Produkt behandelt, und es kann vorkommen, daß das Korrekturprogramm den MMPM nicht für eine Fehlerkorrektur vorsieht. Das kann u.U. auch beim *FP 38* für Warp 3 auftreten. Es gibt dazu allerdings eine kleine Batchdatei von IBM, die dem Fixpack beiliegt und das Problem behebt. Wie und wann man sie verwenden sollte, finden Sie in der Readme-Datei zum *Fixpack*.

Übrigens läßt sich der *Build*- oder *Servicelevel* des Systems auch mit dem Befehl *ver* abfragen. Geben Sie dazu auf der Kommandozeile ein:

ver/ r

Der Parameter /r bewirkt dabei die Ausgabe der sogenannten Überarbeitungsversion. Für ein Warp 4-System, auf dem noch kein *Fixpack* installiert wurde, sieht die Ausgabe so aus:

Version 4.00 des Operating System/2 wird verwendet.
Überarbeitungsversion 9.023

Die Überarbeitungsversion 9.025 steht dabei für *Fixpack 1*. Dann wird einfach aufsteigend weitergezählt, d.h. wurde das aktuelle *Fixpack 10* installiert, wird

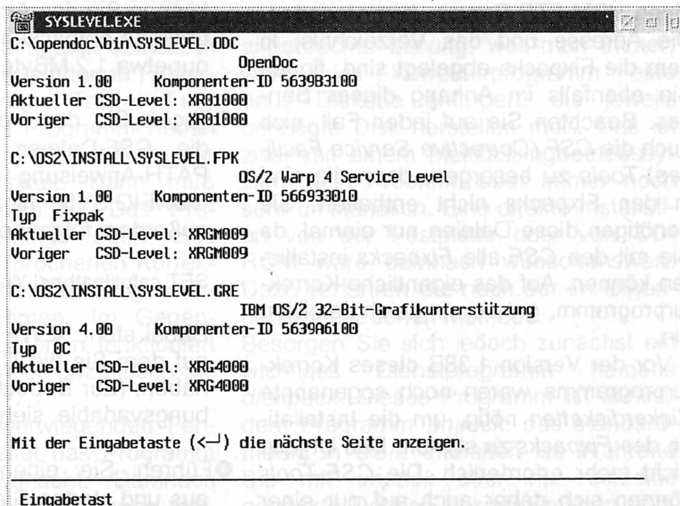


Abb. 2: Anzeige des OS/2 Service-Levels eines Warp 4 Systems, auf das FP 9 installiert wurde. Diese Informationen stehen in der Datei SYSLEVEL.FPK.

die Überarbeitungsversion 9.034 ausgegeben.

Übrigens konnte man vor *FP 32* für Warp 3 bzw. *FP 4* für Warp 4 nur mit dem Befehl *ver /r* herausfinden, welches *Fixpack* installiert wurde. Jetzt zeigt auch *syslevel* an, welches *Fixpack* aufgespielt wurde. Die Informationen dazu befinden sich in der Datei *syslevel.fpk* im Verzeichnis *\os2\install*.

Den bereits erwähnten *APAR* finden Sie übrigens in jedem *Fixpack* als Datei *README2*. Informationen über das *Fixpack* mit Hinweisen zur Installation in der Datei *README.1ST*.

Wo bekommt man die Fixpacks?

Es gibt unterschiedliche Wege, sich das jeweils aktuellste *Fixpack* zu besorgen. Zum einen wären da die *IBM Service-Pac-CDs*, die wir bereits in Bd. 1 vorstellten. Man bekommt Sie über OS/2-Softwarehändler (z.B. Deckarm & Co.) oder direkt von IBM (Adressen in Bd. 1 und im Anhang). Man kann sie sich aber auch direkt aus dem Internet über einen IBM-FTP-Server herunterladen. Die Adresse und das Verzeichnis, in dem die Fixpacks abgelegt sind, finden Sie ebenfalls im Anhang dieses Bandes. Beachten Sie auf jeden Fall, sich auch die *CSF (Corrective Service Facilities)-Tools* zu besorgen, denn die sind in den Fixpacks nicht enthalten. Sie benötigen diese Dateien nur einmal, da Sie mit den *CSF* alle *Fixpacks* installieren können. Auf das eigentliche Korrekturprogramm, gehen wir gleich detailliert ein.

Vor der Version 1.38B dieses Korrekturprogramms waren noch sogenannte *Kickerdisketten* nötig, um die Installation des *Fixpacks* zu starten. Das ist jetzt nicht mehr erforderlich. Die *CSF-Tools* können sich daher auch auf nur einer Diskette, auf CD oder der Festplatte befinden. Wir empfehlen Ihnen, entweder eine eigene *Fixpack-CD* zu erstellen oder die *CSF-Tools* auf der Festplatte zu installieren, um sie für spätere Fixpackinstallationen verwenden zu können. Gehen Sie dazu so vor:

❶ Laden Sie die *CSF-Tools* via FTP aus dem Internet. Die Adresse lautet:

ftp://service.boulder.ibm.com/ps/products/os2/fixtool/german

Sie finden in diesem Verzeichnis eine Datei namens *cgs140.exe*, welche die aktuelle Version 1.40B der *CSF* beinhaltet. Diese Datei laden Sie vom Server. Die Datei *cgs140.txt* benötigen Sie nicht. Sie ist identisch mit der Readme-Datei im Programmarchiv.

❷ Die Datei *cgs140.exe* ist ein selbstentpackendes ZIP-Archiv. Erstellen Sie ein Verzeichnis, am besten namens *csftools* auf Ihrer Betriebsystempartition, und entpacken Sie die Datei in diesem Verzeichnis. Alternativ dazu können Sie das Archiv auch auf einer Diskette entpacken (die *CSF* belegen nur etwa 1,2 MByte).

❸ Fügen Sie das Verzeichnis, in dem die *CSF*-Dateien liegen, zu Ihrer PATH-Anweisung in der CONFIG.SYS hinzu. Nehmen Sie außerdem folgenden Eintrag vor:

SET csfutilpath=<LW>:\csftools

Dabei steht <LW> für das Laufwerk, auf dem Sie die *CSF-Tools* installiert haben. (Zur Bedeutung dieser Umgebungsvariable siehe unten).

❹ Führen Sie einen Systemabschluß aus und starten Sie das System neu, um die Änderungen wirksam werden zu lassen.

Das Korrekturprogramm

Die Installation des *Fixpacks* nehmen Sie mit einem PM-Programm namens *service.exe* vor, das in den *CSF* enthalten ist. Diese Anwendung gibt es schon recht lange, und sie wurde auch im Laufe der Zeit erheblich verbessert. An der Oberfläche hat sich zwar nicht sehr viel getan, aber ansonsten ist das Programm viel zuverlässiger geworden. Nur die Online-Hilfe hätte man an bestimmten Stellen auch ganz fortlassen können, weil Sie zu Fehlermeldungen meist nur ausgibt, daß zu einem jeden Fehler ein entsprechender Text in der Protokolldatei vermerkt ist. Nur hilft dieser Text nicht weiter, da er den Fehler nur meldet und in der Meldung auf Umstände verweist, die zur Klärung des Problems auch nicht sehr oft weiterhelfen. Ein Klick auf den Button *Hilfe* in der erscheinenden Dialogboxen bringt Sie wieder an den Anfang: Bitte schauen Sie in die Protokolldatei oder in die Datei Readme. Die Online-Hilfe des Programms bedürfte also einer dringenden Überarbeitung.

Ein weiterer Schwachpunkt besteht in der praktisch nicht vorhandenen Fehlertoleranz des Programms. Tritt ein Fehler auf, so bietet das Programm neben einem Blick in die Protokolldatei nur eine Abbruchmöglichkeit. Dann muß man von vorne beginnen. Das Programm bietet allerdings auch eine Funktion, einen unterbrochenen Korrekturprozeß beim erneuten Programmstart wiederaufzunehmen. Im Gegensatz zu früheren Versionen funktioniert dies jetzt auch. Hinweise dazu lesen Sie weiter unten.

Sofern keine schwerwiegenden Fehler auftauchen, arbeitet das Programm allerdings wie gewünscht. Demnach sollte man einige Vorbereitungen treffen, um Fehler von vornherein zu vermeiden. Auf die einzelnen Fehlermeldungen, die auftreten können, gehen wir noch gesondert ein, so daß Sie wissen, was im Falle eines Falles zu tun ist.

Wie man die Installation vorbereitet

Um ein *Fixpack* zu installieren, kann man auf verschiedene Arten vorgehen. Eine ist, die einzelnen Diskettenimages mit Hilfe des Programmes *loaddskf.exe* auf Disketten zu entpacken. Anschließend startet man das Fehlerkorrekturprogramm aus den *CSF*, wählt im erscheinenden Dialog als Laufwerk das Laufwerk A: und legt je nach Aufforderung die zuvor erzeugten Disketten ein. Das ist aber etwas umständlich. Einmal dauert das Erzeugen der Korrekturdisketten sehr lange, und auch das ständige Wechseln der Disketten stört, zumal die Tage, in denen man als Diskjockey tätig war, eigentlich im Zeitalter der CD-ROMs vorbei sein sollten (v.a. wenn man bedenkt, daß man meist mit über 15 Disketten zu tun hätte). Auch die einzelnen Disketten auf einem virtuellen Floppylaufwerk zu erstellen, ist keine angebrachte Lösung, weil man immer, wenn das Korrekturprogramm eine neue Diskette anfordert, die jeweils benötigte Disk herstellen muß. Das ist zwar mit einem Multitaskingbetriebssystem kein Problem, aber immer noch sehr umständlich. Eine direkte Installation von der Festplatte oder von CD-ROM wäre demnach wünschenswert. Dazu verfahren Sie nach der im folgenden angegebenen Methode.

Besorgen Sie sich jedoch zunächst ein wichtiges Dienstprogramm namens *diunpack*. Dieses Programm ist ähnlich dem Programm *unpack*, das standardmäßig in OS/2 enthalten ist. Während Sie mit *unpack* aber nur einzelne gepackte Dateien dekomprimieren können, haben Sie mit *diunpack* die Möglichkeit, ein ganzes Diskettenimage in

ein Verzeichnis zu entpacken, was mit *loaddskf* leider nicht möglich ist. Dieses Programm erfordert ein Diskettenlaufwerk, um ein Diskimage zu dekomprimieren. Sie müßten hier wieder ein virtuelles Diskettenlaufwerk haben, das im Speicher angelegt wird, die Diskettenabbilder mit *loaddskf* entpacken und vom virtuellen Floppy in ein Verzeichnis kopieren. Mit *diunpack* kann man sich dieses Procedere sparen. Das Programm anzuwenden, ist ganz einfach, wir werden den Gebrauch im folgenden zeigen. Das Tool finden Sie wie immer zum Download auf unserer Homepage (Adresse siehe im Anhang).

Anschließend brauchen Sie natürlich noch die Diskettenimages. Diese können Sie sich vom bereits oben genannten IBM-Server herunterladen. Danach verfahren Sie wie folgt:

❶ Positionieren Sie *diunpack.exe* in ein Verzeichnis, das in der PATH-Variable in der CONFIG.SYS eingetragen ist, am besten wieder c:\os2\apps.

❷ Kopieren Sie die einzelnen Diskettenabbilder in ein temporäres Verzeichnis, z.B. c:\tmp.

❸ Erzeugen Sie im gleichen Verzeichnis ein Unterverzeichnis, z.B. namens ..\fp4009. Darin werden die Dateien aus den Diskettenimages abgelegt.

❹ Rufen Sie danach *diunpack* mit folgenden Parametern auf:

```
diunpack c:\tmp\xrgm009a.dsk -d
                        h:\fp4009
```

Damit wird *diunpack* angewiesen, das Diskettenimage *xrgm009a.dsk* im Verzeichnis *h:\fp4009* zu entpacken. Die Dateinamenerweiterung sollten Sie laut der Programminformation auch weglassen können. Dann quittiert das Programm jedoch den Dienst mit der Meldung *"Die angegebene Einheit oder Datei kann vom System nicht eröffnet werden."* Fügen Sie also die Erweiterung immer hinzu.

❺ Verfahren Sie mit allen Diskettenabbildern so.

Die Vorbereitungen zur Fehlerkorrektur sind nun fast abgeschlossen. Sie müssen nun das Dienstprogramm *service* starten. Dazu könnten Sie an der Kommandozeile einfach *service* eingeben, wenn Sie sich an die oben angegebene Installationsanleitung gehalten haben. Das Programm würde daraufhin eine Dialogbox anzeigen, in der Ihnen ein Disketten- oder CD-ROM-Laufwerk, das die korrigierten Dateien enthält, angeboten wird, bevor das System zur Fehlerkorrektur überprüft wird (s. Abb. 3).

Der ganze Trick ist, vor dem Aufruf von *service* zwei Umgebungsvariablen zu setzen. Schreiben Sie sich am allerbesten eine kleine Batchdatei, die das bewerkstelligt. Diese sähe dann für unser Beispiel so aus:

```
@ECHO OFF
ECHO Starte Fehlerberichtigung...
set csfutilpath=%1
set csfcdromdir=%2
start %1\service.exe
exit
```

Die erste Umgebungsvariable *csfutilpath* haben wir schon erwähnt. Haben Sie diese schon in Ihrer CONFIG.SYS gesetzt, so können Sie die SET-Anweisung aus der Batchdatei fortlassen. Die Variable steht für *Corrective Service Facility Utility Path* und gibt den Pfad an, in dem sich das Programm *service* mit den zur Korrektur notwendigen Dateien befindet. Die zweite Variable *csfcdromdir* steht für *Corrective Service*

Facility CD-ROM Directory und zeigt auf das Verzeichnis in dem sich die entpackten (!) Dateien zur Korrektur befinden. Damit weiß das Korrekturprogramm alles, was nötig ist, um die Korrektur durchzuführen. Eine andere Möglichkeit, *service* zu initialisieren, gibt es übrigens nicht. Das Programm akzeptiert leider keine Kommandozeilenparameter, über die entsprechenden Verzeichnisse angeben könnte.

Öffnen Sie also eine Kommandozeile, setzen Sie die Umgebungsvaribalen und starten erst dann *service*, wird das Programm wie gewünscht arbeiten. Wenn Sie die oben angegebene Batchdatei unter *startfix* abspeichern, sähe der Start der Fixpackinstallation so aus:

```
startfix c:\csftools h:\fp4009
```

Damit wird angenommen, die *CSF-Tools* befinden sich im Verzeichnis *\csftools* auf Laufwerk *c:* und die Fixpackdateien auf Laufwerk *h:* im Verzeichnis *\fp4009*. Die beiden Umgebungsvariablen werden initialisiert, und *service* wird gestartet. Es durchsucht das System dann sofort nach korrigierbaren Produkten. Was dann zu tun ist, lesen Sie im Abschnitt *Die Installation*.

Mit dieser Methode haben Sie die Fehlerkorrektur bedeutend schneller beendet als mit einzelnen Disketten. Sie können das Verzeichnis mit den entpackten Korrekturdateien natürlich auch auf der Festplatte belassen. Allerdings nehmen die dekomprimierten Diskettenabbilder eine ganze Menge Platz weg, etwa 1,2 MByte/ Image. Bei 15 Disketten kommen da bereits an die 18 MByte zusammen, und da die alten Dateien in einem Archiv zur Wiederherstellung des alten *CSD-Levels* vor der Korrektur gespeichert werden, sind es etwa noch einmal so viel. Hier lohnt sich die Anschaffung eines CD-Recorders erneut: Man kann sich eine ganze Reihe *Fixpacks* aufspielfertig auf eine CD brennen.

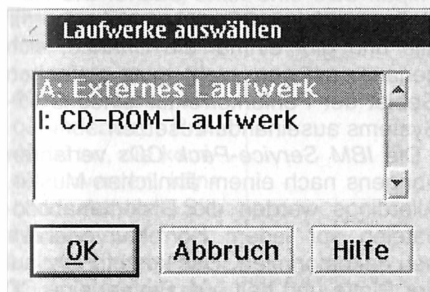


Abb. 3: Die Dialogbox, die service.exe anzeigt, wenn die Umgebungsvariable csfcdromdir nicht gesetzt ist.

Wie gehen Sie zur Herstellung einer solchen CD am besten vor? Überlegen Sie sich zunächst den Aufbau. Sie sollten den Verzeichnissen aussagekräftige Namen geben. Einigen wir uns für dieses Beispiel auf fp<x>0<y>, wobei x für die OS/2-Version und y für die FP-Nummer steht. Wir möchten im folgenden für die *FPs 38* für Warp 3 und *9* für Warp 4 eine CD erstellen. Dazu legen wir uns diese Verzeichnisse an:

<LW>:\fp3038
<LW>:\fp4009

Außerdem benötigen wir noch ein Verzeichnis, in dem das Korrekturprogramm abgelegt wird; dann braucht es nicht auf der Festplatte zu residieren, und eine Installation der *CSF-Tools* auf neuen Rechnern wird überflüssig. Weil wir das gleiche Programm für beide *FPs* verwenden können, erzeugen wir nur ein Verzeichnis namens *csftools*:

<LW>:\csftools

In diese Verzeichnisse entpacken wir direkt die jeweiligen Diskettenimages, wie es bereits beschrieben wurde.

Damit wäre die CD im wesentlichen fertiggestellt. Was nun noch fehlt, ist ein kleines Programm, um die Installation der auf der CD enthaltenen *FPs* in Gang zu bringen. Dazu kann man natürlich -wie wir es für die *OS/2 Only!-CD* und für *Xelia* gemacht haben- ein komfortables PM-Programm mit einer Datenbank und anderen Funktionen entwickeln. Die winzige Batchdatei, auf die Sie weiter oben bereits gestoßen sind, erfüllt allerdings genau den gleichen Zweck, wenn man sie etwas modifiziert:

```
@ECHO OFF
ECHO Starte Fehlerberichtigung...
set csfutilpath=%1:\csftools
set csfcdromdir=%1:\%2
start %1\csftools\service.exe
exit
```

Dieses Programm haben wir (wie oben) *startfix.cmd* genannt und im Hauptverzeichnis der CD abgelegt. Wenn Sie später einmal weitere *FPs* auf die CD kopieren, brauchen Sie dem Stapelverarbeitungszwerg nur die entsprechenden Parameter übergeben. So halten Sie die CD nur durch Anlegen eines neuen Verzeichnisses immer aktuell. Die Anwendung des Batches ist simpel. Nehmen wir an, ein *Warp 4-FP 9* befindet sich im Verzeichnis *\fp4009* und das Programm *service* mit den *CSF-Tools* im Verzeichnis *\csftools* auf Ihrer CD, die Sie in das CD-ROM-Laufwerk mit dem Buchstaben *i:* eingelegt haben. Um das *Fixpack* zu installieren, öffnen Sie eine Kommandozeile und geben ein:

startfix i fp4009

Und schon wird das Korrekturprogramm gestartet. Einfacher kann das Korrigieren des Systems nicht sein. Wenn Sie ein neues *Fixpack* wie oben beschrieben auf die CD kopiert haben, z.B. in das Verzeichnis *\fp4010*, brauchen Sie nur:

startfix i fp4010

einzugeben, und alles läuft wie gewünscht. Natürlich sollten Sie den Batchzwerg noch erweitern, eine Fehlerkorrektur der Parameter wegen hinzufügen oder die Datei als Ausgangspunkt für Ihre eigenen Softwareprojekte benutzen.

Die hier vorgestellte Vorgehensweise ist übrigens auch für kleinere Netzwerke praktisch. Das Programm *service* dient eigentlich zum Korrigieren von Einzelplatzsystemen. In Netzen arbeitet man mit *fservice* und erstellt zur Durchführung der Korrektur Antwortdateien. Sie können sich dieses Procedere mit einer eigenen *Fixpak-CD* ersparen, indem Sie die CD in das Laufwerk eines Hosts einlegen und von allen anderen darauf zugreifen. Die Erlaubnis auf das Laufwerk vorausgesetzt, können Sie so übers Netz die einzelnen Hosts korrigieren. Wir haben es mit einem kleinen *Xelia-Netz* ausprobiert und waren zufrieden[1]. Auch für Notebooks sind die an die eigenen Bedürfnisse angepaßten *Fixpak-CDs* eine feine Sache. Die Herstellung solcher CDs lohnt sich also für alle und gibt einmal die Chance, sich genauer mit dem nicht ganz einfachen Gebiet der Fehlerkorrektur eines OS/2-Systems auseinanderzusetzen.

Die *IBM Service-Pack CDs* verfahren übrigens nach einem ähnlichen Muster. Allerdings werden die Diskettenabbilddateien vor jedem Korrekturverfahren neu dekomprimiert. Das kostet Platz auf der Platte und hält auf. Einmal eine CD

1 Xelia-Netzwerke sind reine TCP/IP-Netzwerke. Wir testeten unsere CD bislang *nur* mit diesem TCP/IP-Netz.

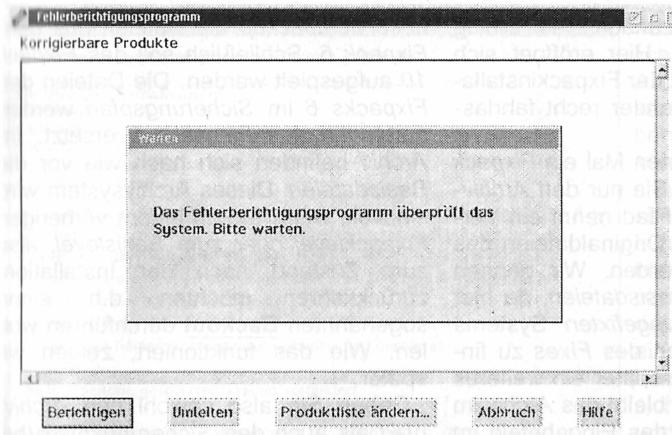


Abb.4: *Service* sucht nach korrigierbaren Produkten

hergestellt, erspart weitere Mühe und Arbeit. Die *Fixpacks* stehen jederzeit bereit zum Aufspielen zur Verfügung.

Die Installation

Wenden wir uns nun dem eigentlichen Installationsprozeß zu, um auf mögliche Fehlerquellen eingehen zu können. Haben Sie das Programm *service* erfolgreich gestartet, so beginnt das Programm sofort damit, das System nach *syslevel*-Dateien zu durchsuchen, um den *Buildlevel* des Betriebssystems zu bestimmen. Sie sehen dann ein Programmfenster wie in *Abbildung 4*. Hat das Programm den Suchprozeß beendet, zeigt es in einer Liste alle gefundenen Produkte an, die korrigiert werden können. Achten Sie hierbei auf den vollständigen Pfad von *syslevel.os2*, der vom Programm angezeigt wird (s. *Abb. 5*). Haben Sie auf Ihrem System neben Warp 4 auch Warp 3 oder OS/2 2.1 installiert, werden diese Systeme ebenfalls als korrigierbar angezeigt. Beachten Sie also, welches System Sie fixen möchten, und selektieren Sie nur dieses System. Behalten Sie ferner im Hinterkopf, für Warp 3 auch den MMPM zur Korrektur hervorzuheben, da diese Systemkomponente unter Warp 3 einzeln aufgeführt wird.

Sie sehen in den beiden Abbildungen am unteren Fensterrand des Programms neben den Buttons *Abbruch* und *Hilfe* auch die Schaltflächen *Berichtigen*, *Umleiten* und *Produktliste ändern*. Wir werden uns zunächst mit dem Button *Berichtigen* beschäftigen. Wenn Sie nach Auswahl des Systems, das korrigiert werden soll, auf diesen Button

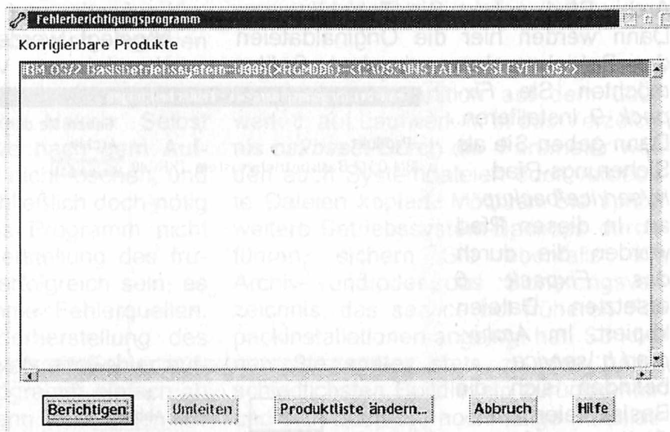


Abb.5: Programmfenster nach Abschluß des Suchprozesses

drücken, erscheint zunächst ein Dialog wie in *Abbildung 6*. Hier eröffnet sich der wichtigste Punkt der Fixpackinstallation, den viele Anwender recht fahrlässig übergehen.

Wenn Sie zum ersten Mal ein *Fixpack* installieren, können Sie nur den *Archivpfad* angeben. Der Pfad nennt ein Verzeichnis, in das die Originaldateien des Systems kopiert werden. Wir nennen Sie im folgenden *Basisdateien*, da hier die Dateien des *ungefixten* Systems nach dem Aufspielen des *Fixes* zu finden sind. Wenn Sie später ein weiteres *Fixpack* installieren, bleibt das *Archiv* im alten Zustand, und das Eingabefeld im Dialog wird grau unterlegt. Um die Dateien des bereits korrigierten Systems zu archivieren, geben Sie im Feld *Sicherung*, das dann für Eingaben bereit steht, einen *Backup-Pfad* an. Nehmen Sie später noch eine Fixpakinstallation vor, werden die Dateien des bereits korrigierten Systems ebenfalls in diesem Pfad abgelegt. Die korrigierten Dateien der vorherigen Installation werden also ersetzt. Dazu ein Beispiel: Sie möchten das *Fixpack 6* auf einem Warp 4-System installieren, das zuvor noch nicht korrigiert wurde. Als Archiv-Pfad geben Sie *h:/service* an. Dann werden hier die Originaldateien des Betriebssystems abgelegt. Später möchten Sie *Fixpack 9* installieren. Dann geben Sie als Sicherungs-Pfad *h:/service/backup* an. In diesen Pfad werden die durch das *Fixpack 6* ersetzten Dateien kopiert. Im *Archivpfad h:\service* befinden sich die Basisdateien; im *Sicherungspfad h:\service\backup* die Dateien aus dem *Fixpack 6*. Schließlich soll das *Fixpack 10* aufgespielt werden. Die Dateien des *Fixpacks 6* im *Sicherungspfad* werden durch die des *Fixpacks 9* ersetzt. Im *Archiv* befinden sich nach wie vor die *Basisdateien*. Dieses Archivsystem wird sinnvoll, wenn Sie zu einem vorherigen *Fixpacklevel* oder zum *Basislevel*, also zum Zustand nach der Installation, zurückkehren möchten, d.h. einen sogenannten **Backout** durchführen wollen. Wie das funktioniert, zeigen wir später.

Geben Sie also sowohl den *Archivpfad* als auch den *Sicherungspfad* (bei späteren Installationen) an. Beachten Sie dazu ferner:

1. Die beiden Verzeichnisse müssen nicht existieren, bevor Sie *service* starten, da das Korrekturprogramm diese selbständig anlegt.

2. Löschen Sie die von *service* erstellten Archive nicht. Wenn Sie es tun, ist ein späterer *Backout* in keinem Falle mehr möglich.

3. Stellen Sie sicher, daß auf dem Laufwerk, auf dem die Archive abgelegt werden, genügend Speicherplatz zur Verfügung steht (wir

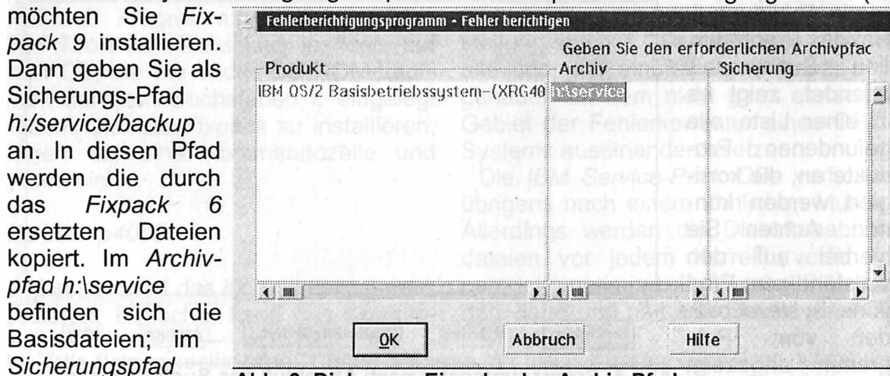


Abb. 6: Dialog zur Eingabe des Archiv-Pfades

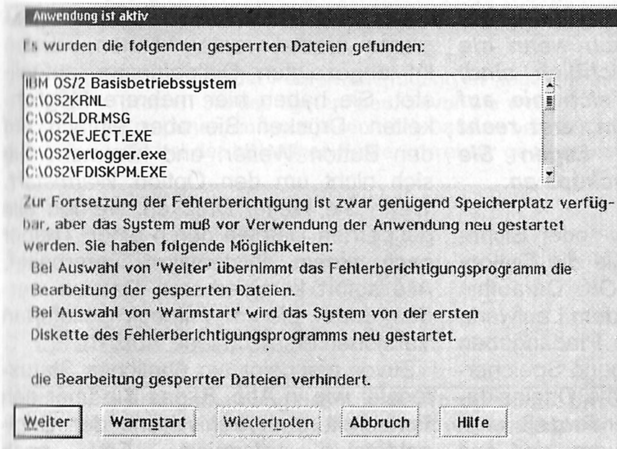


Abb. 7: Anzeige gesperrter Dateien nach Überprüfung des zur Verfügung stehenden Speicherplatzes

empfehlen etwa 30 MByte, besser mehr). Sie ersparen sich spätere Probleme, wenn Sie die Archive von vornherein auf einer Partition anlegen, auf der genügend freier Speicherplatz vorhanden ist.

Ein *Backout* ist immer dann nötig, wenn das installierte *Fixpack* Schwierigkeiten hervorruft, wenn also bestimmte Funktionen nicht mehr verfügbar sind bzw. die Systemstabilität beeinträchtigt ist. Machen Sie also von den Archiven Gebrauch.

Soweit hört sich alles recht gut an, jedoch gibt es einen Haken: Selbst wenn Sie die Archive nach dem Aufspielen des *Fixpacks* nicht löschen, und ein *Backout* sollte schließlich doch nötig werden, so muß das Programm nicht unbedingt bei der Herstellung des früheren *Servicelevels* erfolgreich sein, es gibt hier unangenehme Fehlerquellen. Wenn bei der Wiederherstellung des vorherigen *Servicelevels* ein Fehler auftritt, so bricht das Programm einfach ab (woran das liegen kann, betrachten wir später). Eine Wiederholung des Prozesses ist nicht zwar möglich, doch ist das Archiv beschädigt, kann es nicht mehr benutzt werden. Sie können in einem solchen Falle also nicht zum vorherigen Level zurückkehren. Was bleibt, ist meist nur eine Neuinstallation des Betriebssystems, eine andere Möglichkeit gibt es nicht. Um sich dies zu ersparen, und auch um weiteren Schwierigkeiten vorzubeugen, legen Sie sich nach der Installation und Konfiguration von OS/2 am besten ein Backup Ihrer Betriebssystempartition an, um den ursprünglichen Zustand des Systems jederzeit wiederherstellen zu können. Ein Restoring ist immer schneller durchführbar als eine komplette Neuinstallation und v.a. Neukonfiguration von OS/2. Benutzen Sie zum Anlegen des Backups am besten den Befehl *xcopy*:

```
xcopy c:\* h:\os2base /s /e /r /v /o /t /h
```

Damit »spiegeln« Sie die gesamte Betriebssystempartition auf dem Laufwerk *c:* auf Laufwerk *h:* in das Verzeichnis *os2base*. Durch die Parameter werden auch Systemdateien oder verdeckte Dateien kopiert. Möchten Sie später weitere Betriebssystem-Backups durchführen, sichern Sie ebenfalls das Archiv- und/oder das Sicherungsverzeichnis, das *service* bei früheren Fixpackinstallationen angelegt hat. So können Sie später stets zu den unterschiedlichsten Buildleveln zurückkehren und sich auch die nochmalige Installation bestimmter Fixpacks sparen.

! *Unterschätzen Sie mögliche Fehlerquellen nicht, auch wenn sie nicht gleich offensichtlich sind, und verlassen Sie sich nie auf irgend ein Programm, erst recht nicht auf* service. *Legen Sie daher regelmäßig Backups an.*

Nach Angabe des Archiv- oder Sicherungspfades schließen Sie die Dialogbox durch Drücken auf *OK*. Daraufhin überprüft *service*, ob auf dem Laufwerk, das Sie im Dialog in den Pfadangaben angegeben haben, genügend Speicherplatz vorhanden ist, um die Originaldateien zu speichern. Dieser Prozeß kann mitunter recht lange dauern, und das System reagiert nicht auf Tastatur- oder Mauseingaben. Stören Sie das Programm auch nicht, sondern haben Sie etwas Geduld. Nach der Überprüfung erscheint ein Fenster wie in *Abb. 7*.

Das Programm hat während der Speicherplatzberechnung das System nach den korrigierbaren Dateien durchsucht. Alle gesperrten Dateien, das sind solche, die gerade benutzt werden und daher nicht durch das Korrekturprogramm ersetzt werden können, werden im angezeigten Dialogfenster aufgelistet. Sie haben hier mehrere Möglichkeiten. Drücken Sie aber einfach auf den Button *Weiter*, und kümmern Sie sich nicht um den Option *Warmstart*. Wenn Sie *Weiter* Drücken, werden alle gesperrten Dateien durch einen Treiber nach einem Systemstart verarbeitet. Alle sofort korrigierbaren Dateien werden durch die im *Fixpack* gelieferten Versionen ersetzt (siehe *Abb. 8*).

Zuvor erscheint ein ähnliches Statusfenster wie in *Abb. 8*, das Sie über den Fortschritt des Archivierens der Originaldateien informiert. Erst nach Abschluß des Archivierens werden die Betriebssystemdateien aktualisiert, wie es die Abbildung zeigt. Kann eine Datei nicht sofort aktualisiert werden, wird die Korrektur verzögert. Die Datei wird dann in eine Liste eingetragen, die von dem bereits erwähnten Treiber abgearbeitet wird. Wurden schließlich alle Dateien korrigiert, bzw. die gesperrten Dateien vom Programm für die spätere Verarbeitung vorgemerkt, quittiert *service* den erfolgreichen Prozeß mit einer Fehlermeldung, die keine ist. *Abbildung 9* zeigt das kleine Fenster, dessen Titelleiste man sich lieber nicht durchlesen sollte. Sie können *service* nun durch Drücken auf den Button *Ende* beenden. Zuvor kann man noch einen Blick in die Protokolldatei wer-

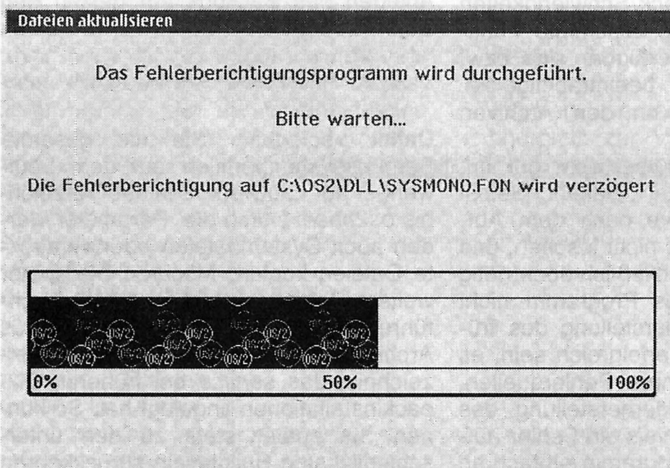


Abb. 8: Nach dem Archivieren der Originaldateien beginnt das Korrekturprogramm mit dem Ersetzen der korrigierbaren Dateien.

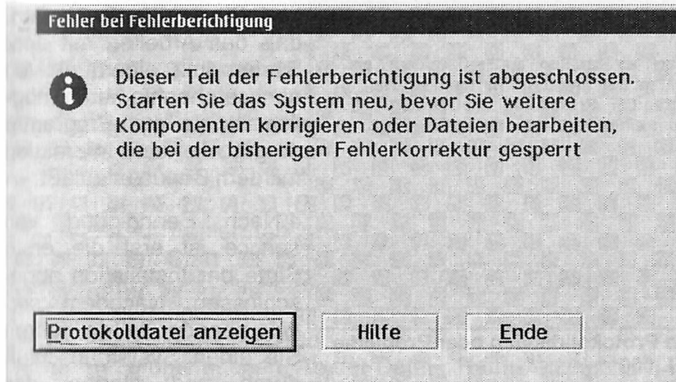


Abb. 9: Die Erfolgsmeldung des Korrekturprogramms. Unverständlicherweise wird sie als Fehlermeldung ausgegeben.

fen. Dort finden Sie alle Meldungen, die während des Korrekturprozesses angezeigt wurden, noch einmal aufgelistet.

Nachdem Sie *service* beendet haben, führen Sie einen Systemabschluß durch und starten das System neu. Ersetzen Sie nach dem Beenden von *service* keine weiteren Dateien. Erst muß der begonnene Korrekturprozeß beendet werden. Dazu hat das Programm in der CONFIG.SYS folgende Einträge vorgenommen:

```
DEVICE=C:\OS2\INSTALL\IBMCSFLK.SYS
            C:\OS2\INSTALL\IBMCSFLK.LST

CALL=C:\OS2\INSTALL\IBMCSFLK.EXE
            C:\OS2\INSTALL\IBMCSFLK.LST
```

Der Treiber *ibmcsflk.sys* und das gleichnamige Programm dienen dem Ersetzen aller gesperrten Dateien durch die neuen Versionen aus dem *Fixpack*. Welche Dateien ersetzt werden müssen, finden Treiber und Programm in der Datei, welche das Programm *service* während des Korrekturprozesses erstellt hat und die unter dem Namen *ibmcsflk.lst* im Verzeichnis *\os2\install* abgelegt wurde. Beim ersten Systemstart nach der Ausführung von *service* wird der Treiber geladen und der Prozeß zum Ersetzen der Dateien gestartet. Ist die Dateiliste abgearbeitet, wird sie gelöscht. Die beiden aufgeführten Einträge in die *config.sys* werden zu diesem Zeitpunkt ebenfalls beseitigt. Schließlich startet das System automatisch durch.

Dieser Prozeß ist erforderlich, da die neuen Versionen der gesperrten Dateien noch nicht geladen wurden (das betrifft vor allem den Betriebssystemkernel). Ein Teil des Systems würde daher mit den Orginaldateien und ein anderer mit den korrigierten Versionen arbeiten. Um dies zu vermeiden, ist ein erneuter Systemstart erforderlich, der von selbst ausgeführt wird. Danach ist die Fehlerkorrektur des Systems abgeschlossen. Der nächste Neustart verläuft ohne Unterbrechung und anschließend läuft OS/2 mit dem neuen *Servicelevel*, wovon man sich mit *syslevel* überzeugen kann.

Tip: Die Dateien IBMCSFLK.SYS und IBMCSFLK.EXE werden bei Beginn des Korrekturprozesses im Verzeichnis \os2\install *abgelegt. Man kann Sie dort für spätere Korrekturprozesse belassen.*

Soweit die Beschreibung des Korrekturprozesses und welche Eingaben Sie zu tätigen haben. Wir werden im folgenden Fehler betrachten, die während der Programmausführung entstehen können.

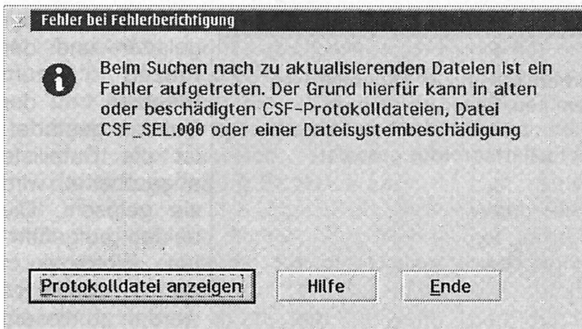


Abb. 10a: Beschädigte Protokolldateien oder Probleme mit CSF_SEL.000

Bevor wir das tun, möchten wir noch einmal einige wichtige Hinweise zusammenfassen und weitere Tips zur Arbeit mit *service* geben:

1. Führen Sie vor der Installation des *Fixpacks* sicherheitshalber ein Backup Ihrer Betriebssystempartition durch. Haben Sie bereits ein *Fixpack* installiert, sichern Sie auch bereits vorhandene Archiv- bzw. Sicherungsverzeichnisse mit, damit spätere *Backouts* nach einem Restoring möglich sind.

2. Beenden Sie alle aktiven Anwendungen, bevor Sie *service* starten.

3. Wenn Sie *service* gestartet haben, stören Sie das Programm während der Ausführung nicht durch andere Systemaktivitäten; auch dann nicht, wenn der Korrekturprozeß sehr lange dauert oder Pausen zwischen den einzelnen Installationsabschnitten auftreten. Benutzen Sie das System nur dann, wenn *service* Sie dazu auffordert. Sie werden feststellen, daß das Arbeiten mit dem System mit einem aktiven *service* ohnehin kaum möglich ist, da das Programm so gut wie keine Interaktion mit dem Benutzer zuläßt.

4. Nach Beendigung von *service* ist erst die erste Stufe der Installation abgeschlossen. Nachdem *service* beendet wurde, führen Sie einen Systemabschluß durch und starten das System neu. Verzichten Sie darauf, es vor dem Neustart zu benutzen.

5. Benutzen Sie *immer* die aktuellste Version der *CSF-Tools*!

Was ist zu tun, ...

Wir wollen nun auf einige Probleme eingehen, die während einer Fixpackinstallation auftreten können. Manchmal nämlich ist deren Ablauf nicht ganz so unproblematisch wie wir ihn gerade beschrieben haben. Sie finden daher im folgenden mögliche Fehlermeldungen und dazu die jeweiligen Vorgehensweisen zur Behebung der einzelnen Fehler. Was also tun Sie...

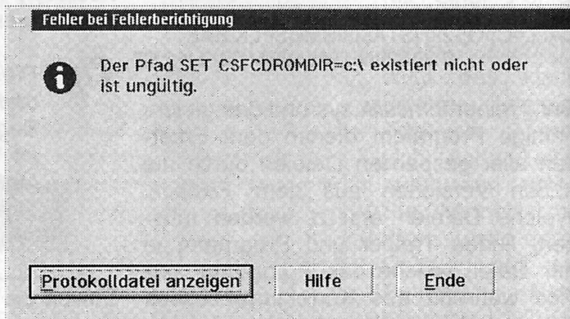


Abb. 10b: Der Pfad mit den Fixpackdateien konnte nicht gefunden werden. Diese Variante der Fehlermeldung wird von den CSF-Tools ab Version 1.40B ausgegeben.

...wenn die CSF-Protokolldateien oder die Datei CSF_SEL.000 beschädigt sind?

Eine Fehlermeldung, die beim Start des Serviceprogramms von Festplatte oder CD-ROM auftritt, finden Sie in *Abb. 10a* und *10b*. Der Fehler taucht bald nach dem Start des Programms auf, kurz nachdem es damit begonnen hat, Ihr System nach korrigierbaren Produkten zu untersuchen. An der Datei namens CSF_SEL.000 oder den Protokolldateien liegt es aber in manchen Fällen nicht, auch wenn das so gemeldet wird. Der Grund liegt bei beiden Fehlermeldungen darin, daß *service* die Fixpackdateien nicht finden kann, d.h. die Variable *csfcdromdir* weist auf ein Verzeichnis, das nicht existiert oder in dem die Fixpackdateien nicht abgelegt sind. Die Fehlermeldung in *Abb. 10b* wird von den *CSF-Tools* ab Version 1.40B ausgegeben; die in *Abb. 10a* vor allem von Versionen vor 1.40B. Die Lösung: Überprüfen Sie, ob *service* auch der richtige Pfad auf das Verzeichnis übergeben wurde, in dem sich das *Fixpack*, genauer die Datei CSF_DISK befindet. Ohne diese Datei geht nämlich gar nichts, was nicht verwundert, weil dort festgehalten ist, welches *Fixpack* überhaupt zur Installation bereitsteht. Da man mit dem gleichen *service* beliebige *Fixpacks* für Warp 3 und Warp 4 aufspielen kann, wird der Sinn dieser Datei schnell klar. Sie befindet sich immer auf der ersten Diskette des *Fixpacks*. Wenn Sie die Diskettenimages z.B. auf Festplatte in das Verzeichnis *\fix* entpackt haben, befindet sie sich dort. Die eigentlichen Fixpackdateien findet man aber in Unterverzeichnissen, die für *service* zunächst nicht relevant sind. Das Programm braucht daher das Hauptverzeichnis, in dem CSF_DISK lokalisiert ist. Sie teilen *service* diesen Pfad wie gesagt mit der Umgebungsvariable *cfscdromdir* mit. Setzen Sie diese auf das Hauptverzeichnis, in das die FP-Dateien samt CSF_DISK entpackt wurden. Anschließend läuft *service* weiter.

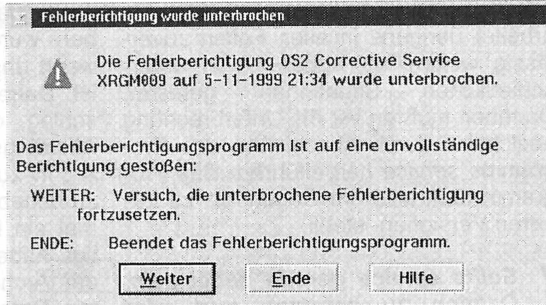


Abb. 11: Installation wurde unterbrochen

...wenn eine Fixpackinstallation unterbrochen wurde?

Wenn eine Fixpackinstallation aus irgend einem Grunde unterbrochen wurde, erhalten Sie beim erneuten Start von service die Fehlermeldung aus *Abb. 11*. In vielerlei Quellen liest man, daß man die Protokolldateien löschen sollte, um ganz von vorne zu beginnen. Das war früher auch nötig. Mit der Version 1.40B der *CSF-Tools* jedoch nicht mehr. Drücken Sie also auf den Button *Weiter*. Das Korrekturprogramm setzt seine Arbeit dann nicht genau an der Stelle fort, an dem es unterbrochen wurde, sondern wiederholt den gesamten Abschnitt der Installation, in dem es sich gerade befand. Trat der Abbruch z.B. während des Archivierens auf, wird der gesamte Archivierungsvorgang wiederholt. Bei einer Unterbrechung des Korrekturprozesses wird mit dem Ersetzen der Dateien ebenfalls von vorne angefangen, unabhängig davon, wie weit der Korrekturprozeß schon fortge-

schritten war. Das Korrekturprogramm arbeitet übrigens in allen Fällen zuverlässig weiter, wir haben es in unterschiedlichen Situationen getestet. Daneben mußten wir die Unterbrechung absichtlich durch Abschießen des Programms *service* herbeiführen. Das Programm läuft also -im Gegensatz zu früheren Versionen- stabil.

! ***Sollte service gerade dabei sein, Dateien zu ersetzen und wird dann abgebrochen, schalten Sie Ihr System nicht aus, sondern starten Sie service sofort erneut. Ein Systemneustart hat zur Folge, daß das System teils mit den Originaldateien, teils mit den bereits gefixten Dateien startet. Das kann zu unvorhersehbaren Effekten führen. In den meisten Fällen startet das System dennoch normal. Das Korrekturprogramm wird seine Arbeit nach einem Neustart ebenfalls wieder an der richtigen Stelle aufnehmen. Wir beobachteten aber, daß wir die Umgebungsvaribalen mit dem Kommando set nicht mehr über die Kommandozeile setzen konnten. In einem solchen Falle nehmen Sie entsprechende SET-Einträge in die CONFIG.SYS vor und starten das System abermals. Anschließend wird service laufen. In allen Testsituationen setzte es seine Arbeit wie gewünscht fort.***

...wenn das Korrekturprogramm nicht auf das Archiv zugreifen kann?
Das Meldungsfenster zu diesem Fehler finden Sie in *Abb. 12*. Der Fehler tritt immer dann auf, wenn der Archivpfad, der *service* einmal übergeben wurde, nicht mehr existiert oder wenn das noch bestehende Archiv leer ist. Daher beherzigen Sie unseren Ratschlag, die Archive nicht zu löschen. Allerdings kann auch ein beschädigtes Archiv zu diesem Fehler führen. Um ihn zu beheben, können Sie, sofern Sie einmal ein Backup des Archives angefertigt haben, bevor es gelöscht wurde, das Archivverzeichnis mit den archivierten Dateien neu anlegen und *service* anschließend noch einmal starten. Liegt jedoch kein Backup des Archives vor, müssen Sie die Protokolldateien des Korrekturprogramms löschen. Sie werden beim Start von *service* automatisch angelegt, sollten sie nicht vorhanden sein. Dann können Sie einen neuen Archivpfad angeben. Um so zu verfahren, suchen Sie im Verzeichnis *\os2\install* nach den Dateien:

LOGF0000.OS2
LOGSTART.OS2
LOGARCH.OS2

Und für eine unterbrochene Fixpackinstallationen unter Warp 3 im Verzeichnis *\os2\mmpm\install*, nach den Dateien:

LOGF0000.MPM
LOGSTART.MPM
LOGARCH.MPM

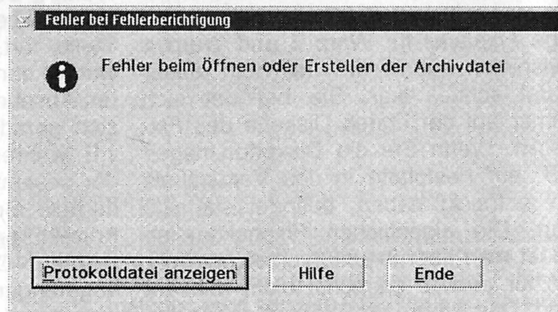


Abb. 12: Kein Zugriff auf das Archiv

sowie nach der Datei CSF_SEL.000 jeweils im Stammverzeichnis des Installationslaufwerkes und *aller anderen startbaren Partitionen des Systems*, und löschen Sie sie (einige dieser Dateien sind schreibgeschützt, entfernen Sie den Schreibschutz vor dem Löschen also mit attrib -r). Sie können nach dem Löschen der Protokolldateien übrigens auch alle noch bestehenden Archiv- und Sicherungsverzeichnisse beseitigen, da sie ohne die zuvor gelöschten Protokolldateien ohnehin nicht mehr verwendet werden können.

Starten Sie nach diesen Schritten *service* erneut, so ist die Angabe eines neuen Archivpfades wieder möglich. Das Programm läuft dann wie gewünscht weiter.

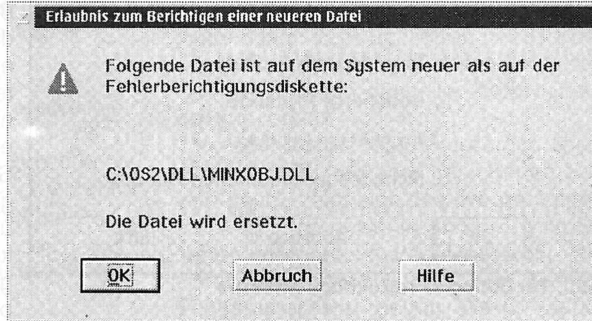


Abb. 13: Erlaubnis zum Ersetzen einer Datei

...wenn eine Datei korrigiert werden soll, die aktueller ist als die im Fixpack?

Sollte auf Ihrem System eine Datei vorhanden sein, die neueren Datums als Ihr Pendant im *Fixpack* ist, ersetzt *service* diese Datei nicht selbständig, sondern bittet mit dem Dialog in *Abb. 13* um Erlaubnis für die Korrektur.

In einem solchen Fall gestatten Sie die Korrektur der Datei *stets* durch Drücken auf *OK*. Das mag vielleicht unsinnig erscheinen, ist jedoch dringend angeraten, da einige Dateien in einem *Fixpack* aus Gründen der Funktionsfähigkeit des gesamten Paketes durch ältere Versionen ersetzt worden sein können. Grundsätzlich ist es immer besser, das *Fixpack* in seiner Gesamtheit aufzuspielen, d.h. die zur Korrektur vorgesehenen Dateien sollten auch ohne Ausnahme ersetzt werden. Allerdings kann dadurch die Funktionalität anderer Softwarepakete beeinträchtigt werden. Derartige Fälle sind uns bislang nur mit der *Java-Unterstützung* bekannt, da auch hier Systemdateien ersetzt werden. Daher empfiehlt es sich, zunächst ein *Fixpack* aufzuspielen und erst dann die entsprechende *Java-Version*. Genaueres dazu erfahren Sie in den Readme-Dateien des jeweiligen *Fixpacks* bzw. der für den Einsatz vorgesehenen *Java-Version*. Beachten Sie die dort gemachten Hinweise sorgfältig. Bis auf diese Ausnahme empfehlen wir jedoch, dem Programm immer die Erlaubnis zur Korrektur von Dateien zu geben.

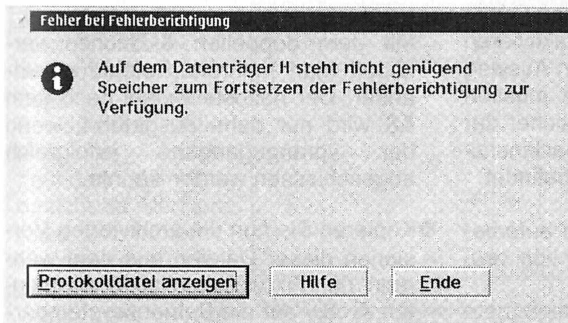


Abb. 14: Nicht genügend Speicherplatz für *service*

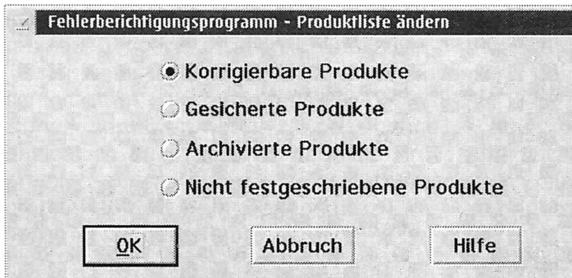


Abb. 15: Dialog »Produktliste ändern«

...wenn nicht genügend Speicherplatz zur Verfügung steht?
Diese Fehlermeldung (s. *Abb. 14*) tritt immer dann auf, wenn Sie ein Archiv- oder Sicherungsverzeichnis angegeben haben, das sich auf einer Partition befindet, auf der nicht mehr genügend Speicherplatz zur Verfügung steht. Sie müssen das Programm zunächst an dieser Stelle beenden. Anschließend können Sie nach dem Neustart von *service* ein anderes Verzeichnis in dem bereits bekannten Dialog angeben. Ein Löschen der Protokolldateien ist in diesem Fall allerdings nicht nötig.

Wenn Sie jedoch tatsächlich Speichernot haben, ist es auch möglich, die Verzeichnisse auf einem wechselbaren Datenträger anlegen zu lassen, etwa auf einer ZIP-Diskette oder sogar auf einer CD-R (mit dem RSJ-Dateisystem ist dies sehr einfach, und da man die Archivierung bzw. Sicherung von Dateien durch *service* nicht deaktivieren kann, hat man zumindest einen Ausweg für derartige Situationen). Sie müssen dann nur darauf achten, daß sicher der Datenträger bei erneuten Fixpackinstallationen im richtigen Laufwerk befindet.

...wenn das System nach dem automatischen Neustart immer wieder von neuem durchstartet?
Dieser interessante Effekt begegnete uns nach der Installation des *Fixpacks* *36* für Warp 3: Nachdem die gesperrten Dateien verarbeitet worden sind, startete der Rechner wie gewohnt neu. Nach dem Laden der ersten Treiber wurde das System jedoch abermalig gestartet, daraufhin wieder usw. Diese Endlosschleife konnten wir nach langwierigen Versuchen und kompletten Neustinstallationen des Betriebssystems sowie wiederholtem Aufspielen des *Fixpacks* nur durch den Austausch einiger Systemdateien beenden. Sollten Sie einmal mit diesem oder einem ähnlichen Problem konfrontiert werden, versuchen Sie folgende Lösung:

❶ Starten Sie Ihr System von Diskette oder einer Wartungspartition, und wechseln Sie auf die Betriebssystempartition, auf der das *Fixpack* installiert wurde.

❷ Heben Sie die Attribute r, s und h für die beiden Dateien OS2LDR und OS2LDR.MSG auf, und löschen Sie diese anschließend:

```
attrib -r -s -h os2ldr && del os2ldr &&
        attrib -r -s -h os2ldr.msg &&
                        del os2ldr.msg
```

Mit dem doppelten &-Zeichen verknüpft man mehrere Befehle miteinander. Der nächste Befehl nach dem && wird nur dann ausgeführt, wenn der vorangegangene erfolgreich abgeschlossen werden konnte.

❸ Kopieren Sie nun die archivierten Versionen dieser Dateien aus dem während der Fixpackinstallation angelegten Archiv auf die Betriebssystempartition:

```
unpack <LW>:\archive\os2ldr.___
        <BLW>:\os2ldr

unpack <LW>:\archive\os2ldr.ms_
        <BLW>:\os2ldr.msg
```

Dabei ist <LW> das Laufwerk, auf dem die archivierten Originaldateien liegen und <BLW> Ihr Bootlaufwerk.

⓫ Setzen Sie die Dateiatribute wieder:

```
attrib -r -s -h os2ldr &&
        attrib -r -s -h os2ldr.msg
```

und starten Sie das System neu.

Mit dem ausgetauschten **Loader** lief das System wieder normal, so daß wir die Fixpackinstallation vollständig rückgängig machen konnten.

Wie man Backouts durchführt

Wir haben bereits mehrmals den Begriff *Backout* erwähnt und ihn definiert als Wiederherstellung eines früheren Service-Levels. In diesem Abschnitt wollen wir zeigen, wie man einen *Backout* durchführt, sollte dies einmal nötig sein.

Auch hierzu benutzt man das Programm *service*. Nach dem Start drückt man im Programmfenster auf den Button *Produktliste ändern...*, worauf der Dialog aus *Abb. 15* erscheint. Je nachdem, ob man die Basisversion (die sich im Archiv befindet) oder eine gesicherte Version (also ein früheres Service-Level) wiederherstellen will, wählt man in der Dialogbox *Archivierte Produkte* oder *Gesicherte Produkte*. Nach dem Schließen des Dialogs durch Drücken auf *OK* durchsucht *service* das System nach der Archiv- bzw. Backupversion.

Vorausgesetzt, das Programm findet einen archivierten oder gesicherten Service-Level, zeigt sich das Programmfenster von *service* wie in Abb. 16. Wie Sie sehen, durchsucht das Programm alle Partitionen Ihres Systems. Auf unserem Testrechner war neben Warp 4 auch noch Warp 3 installiert. Daher zeigt *service* das Basisbetriebssystem und den MMPM (mit installiertem FP 40) ebenfalls für einen *Backout* an. Standardmäßig sind alle Einträge selektiert. Heben Sie die Selektion für all jene Einträge, die für das gerade aktive System nicht in Frage kommen, auf und drücken Sie dann auf den Button *Zurücksetzen*. Daraufhin erscheint ein Dialogfenster, das *Abb. 17* zeigt.

Unter *Produkt* wird der gerade aktive *Service-Level* angezeigt. Der Bereich *Zielversion* gibt den *Service-Level* wieder, der durch das Zurücksetzen wiederhergestellt wird. In unserem Falle wollen wir die Basisversion des Betriebssystems wiederherstellen.

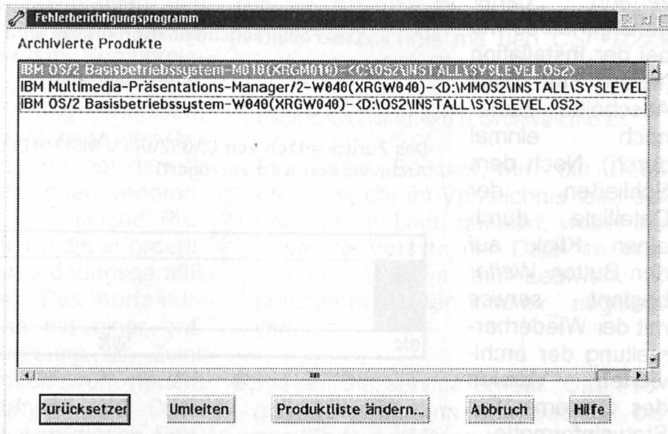


Abb.5: Programmfenster nach Abschluß des Suchprozesses

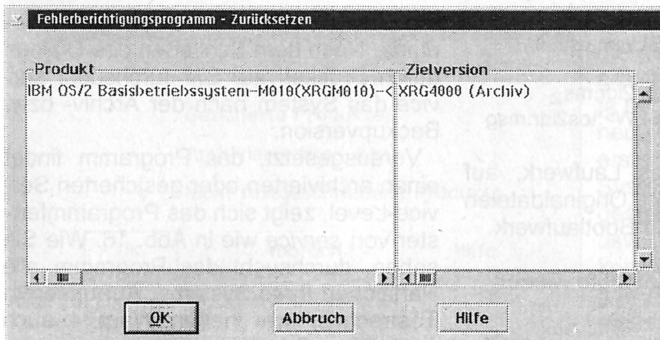


Abb. 17: Dialog für den möglichen Backout

Zuvor installierten wir das *Fixpack 10* auf unser Warp 4-System. Zum Backout müssen wir nun lediglich den Dialog durch drücken auf *OK* schließen. Daraufhin wird der *Backout* vorgenommen.

Der Programmablauf ist dabei im wesentlichen genau der gleiche wie bei der Installation des *Fixpacks*. Auch hier wird nach Schließen des Dialogfensters, das die obige Abbildung zeigt, das System nochmalig überprüft. Dann wird die bereits bekannte Liste der gesperrten Dateien angezeigt, d.h. auch der Deinstallationsprozeß eines *Fixpacks* ist zweiteilig. Man verfährt daher ganz genau so wie bei der Installation (lesen Sie diese Abschnitte daher noch einmal durch). Nach dem Schließen der Dateiliste durch einen Klick auf den Button *Weiter* beginnt *service* mit der Wiederherstellung der archivierten Version des Systems. Die Statusinformationen, die dabei ausgegeben werden, sind *Abb. 18* zu entnehmen.

Sobald der erste Teil des *Backouts* abgeschlossen ist, quittiert dies *service* durch eine Meldung, wie sie *Abb. 19* zeigt. Man bestätigt sie wie bei der Installation des *Fixpacks* und startet das System erneut. Beim Neustart wird der Gerätetreiber zur Verarbeitung gesperrter Dateien geladen und die Liste der gesperrten Dateien abgearbeitet. Nach dieser Prozedur wird der Rechner wie schon bei der Installation neu gestartet. Nach dem Neustart arbeitet OS/2 wieder auf dem *Service-Level*, auf den man das System zurückgesetzt hat, in unserem Beispiel wurde die Basisversion von Warp wiederhergestellt. Man kann sich auch hier leicht mit den Befehlen *ver /r* und *syslevel* vom Erfolg des *Backouts* überzeugen.

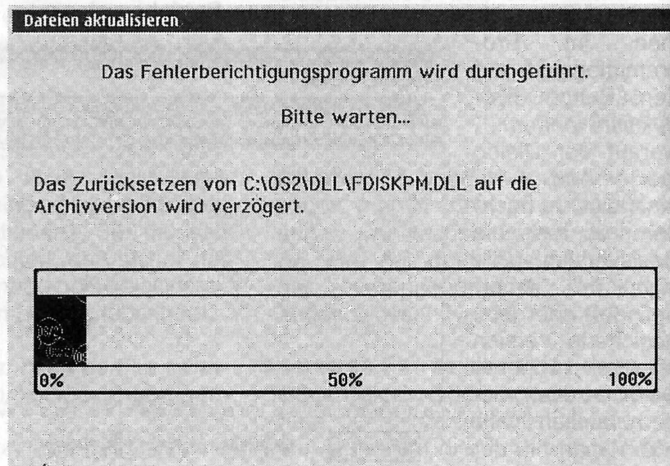


Abb. 18: Verzögerung des Backouts für gesperrte Dateien

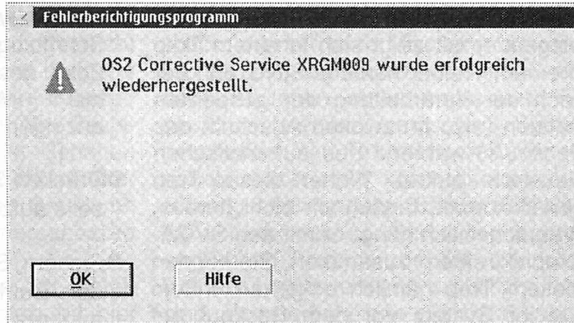


Abb. 19: Meldung zur Wiederherstellung eines früheren Service-Levels, hier auf XRGM009.

Probleme während des Backouts
Schwierigkeiten beim Zurücksetzen auf einen früheren Service-Level gab es v.a. mit Versionen der *CSF-Tools* vor 1.40B. Hier war es stellenweise gar nicht möglich, bei einem unterbrochenen *Backout* den Wiederherstellungsprozeß nochmals aufzunehmen. Jetzt geht das erfreulicherweise. Sollte also einmal aus irgend einem Grunde das Zurücksetzen unterbrochen werden, so können Sie *service* einfach erneut starten (wie schon bei der Installation des *Fixpacks*). Der Wiederherstellungsprozeß beginnt dann noch einmal von vorn. Allerdings können während beiden Abschnitten des *Backouts* unangenehme Fehler auftauchen, die etwas mehr Aufwand erfordern (allerdings treten sie auch recht selten auf). Es handelt sich dabei zunächst um einen Fehler, der während der **CRC-Prüfung** auftritt. Das liegt daran, daß *service* die Originaldateien während der Fixpackinstallation mit dem Programm *pack* (dem Pendant zu *unpack*), komprimiert und beim *Backout* mit *unpack* wieder dekomprimiert. Bei jedem Komprimierung/ Dekomprimierungsalgorithmus wird mit einer *CRC-Prüfung* kontrolliert, ob die dekomprimierte Version der Originalversion entspricht, da bei derartigen Vorgängen Informationen verlorengehen können. Fällt eine solche Prüfung negativ aus, so kann die erforderliche Datei nicht mehr ordnungsgemäß dekomprimiert werden. Das Korrekturprogramm bricht dann mit einer entsprechenden Fehlermeldung ab. Zwar können Sie den Prozeß von neuem starten, aber die komprimierte Datei bleibt natürlich defekt. Um diesen Fehler zu beheben, versuchen Sie folgende Vorgehensweise:

❶ Bestimmen Sie anhand der Fehlermeldung bzw. der Protokolldatei, an welcher Stelle *service* den *Backout* durch die defekte Datei unterbrechen mußte.

❷ Wechseln Sie in das Archivverzeichnis und löschen Sie diese Datei.

❸ Wechseln Sie in das Verzeichnis auf Ihrer Betriebssystempartition, in dem die Originaldatei liegt.

❹ Rufen Sie das Programm *pack* aus dem Verzeichnis mit den *CSF-Tools* auf und packen Sie die Originaldatei:

```
pack c:\os2\cmd.exe h:\archive\cmd.ex_
```

In diesem Beispiel wird die Datei *cmd.exe*, die im Verzeichnis *\os2* auf Laufwerk *c:* liegt, gepackt, wobei die gepackte Version der Datei im Verzeichnis *\archive* auf Laufwerk *h:* unter dem Namen *cmd.ex_* abgelegt wird.

❺ Starten Sie *service* erneut. Das Programm wird dann den *Backout* bis zum Ende ausführen können.

Der zweite Fehler, der weitaus schwerwiegender ist, zeigt sich in einem Trap in einem Treiber namens SINGLE$, der nach der Verarbeitung der gesperrten Dateien (also im zweiten Abschnitt des *Backouts*) während des automatischen Neustarts auftrat. Woher dieser Trap genau kommt, fanden wir nicht heraus. Wahrscheinlich hängt er mit den SVGA-Graphiktreibern zusammen. Sie können diesem Trap nämlich entgehen, indem Sie Ihr System vor dem Backout auf Standard-VGA zurücksetzen. Öffnen Sie dazu einen OS/2-Gesamtbildschirm, geben Sie den Befehl *setvga* ein, und starten Sie das System neu. Danach kann mit dem *Backout* wie angegeben begonnen werden, ohne auf den beschriebenen Fehler zu stoßen.

! ***Setzen Sie das System auf Standard-VGA mit dem Befehl* setvga *zurück, wenn Sie das System auf die Archivversion zurücksetzen wollen. Verfahren Sie sicherheitshalber auch beim Zurücksetzen auf Service-Levels in Sicherungsarchiven so.***

Sollten Sie dies aber einmal vergessen haben, und ist es erst zu spät, können Sie sich eine Neuinstallation des Betriebssystems ersparen, wenn Sie ein Backup Ihrer Betriebssystempartition zur Hand haben. Falls nicht, bekommen Sie Ihren Rechner aber dennoch zum Laufen, wenn Sie von der Archivierungsfunktion der WPS Gebrauch machen. Dieser Lösungsweg verhilft dem System in einem solchen Fall wieder zur Weiterarbeit:

❶ Starten Sie das System durch Drücken der Tasten *Strg+Alt+Entf* neu, und drücken Sie *Alt+F1*, wenn das kleine weiße Rechteck mit dem Schriftzug OS/2 in der linken oberen Ecke des Bildschirms erscheint, um die Wiederherstellungsfunktionen anzeigen zu lassen.

❷ Drücken Sie *F2*, um eine Kommandozeile aufzurufen.

❸ Sichern Sie die Datei *config.sys* von der Betriebssystempartition auf ein anderes Laufwerk. Starten Sie das System anschließend wieder neu, und rufen Sie abermals den Bildschirm mit den Wiederherstellungsoptionen aus.

❹ Wählen Sie jetzt das Originalarchiv nach der Installation zur Wiederherstellung durch Drücken der Taste *0* aus. Daran anschließend werden die Schlüsseldateien des Archives wiederhergestellt. Das System startet nach dem Zurücksetzen auf den Installationszustand normal.

Leider ist Ihre gesamte Konfigurationsarbeit damit zunichte gemacht. Zumindest können Sie mit der gesicherten *config.sys* alle Pfadangaben und Umgebungsvariablen, die von Programmen während der Installation gesetzt wurden, relativ schnell wiederherstellen, so daß »nur« noch die Neukonfiguration der WPS bleibt. Das ist allerdings allemal besser als eine komplette Neuinstallation des Betriebssystems. Daher noch einmal die dringende Empfehlung:

! ***Führen Sie nach der Installation und vollständigen Konfiguration von OS/2 sowie in regelmäßigen Abständen (nach jeder Fixpackinstallation und jeder Neuinstallation von Programmen etc.) Backups Ihrer Betriebssystempartition durch.***

Der Trap nach dem Zurücksetzen auf die Archivversion trat allerdings nicht auf jedem Testsystem auf und übrigens gar nicht, wenn nur auf ein früheres *Service-Level* und nicht auf die Basisversion zurückgesetzt wurde. Behalten Sie die Möglichkeit dieses Fehlers aber immer im Auge, und sorgen Sie durch Sicherungskopien unvorhersehbaren Problemen vor.

Wenn Sie Fehler während der Installation von *Fixpacks* beobachten, die hier nicht wiedergegeben sind, und Lösungen für diese Schwierigkeiten haben, schicken Sie uns bitte Ihre Tips. Besonders Hinweise, sie sich auf das Problem mit dem Trap beim Zurücksetzen auf die Archivversion beziehen, nehmen wir gerne entgegen. Falls es hier andere Vorgehensweisen zur Problembeseitigung gibt, würden wir gerne davon wissen.

Grundsätzlich beachten Sie bitte zur Durchführung eines *Backouts*, daß Sie auch hier die Umgebungsvariablen vor dem Aufruf von *service* wie oben beschrieben setzen müssen, wenn Sie das Programm von der Festplatte ausführen wollen. Dabei muß auch das Verzeichnis mit den Fixpackdateien bestehen, sonst startet *service* nicht korrekt. Als Alternative bietet sich der Start des Programms von Diskette. Auch hier muß man allerdings eine Fixpackdiskette bereithalten. Am einfachsten gestaltet sich die gesamte Arbeit mit Fixpacks mit einer selbst hergestellten CD, wie wir es gezeigt haben. Der Aufruf des kleinen Batchprogramms ist auch für *Backouts* stets derselbe.

Zum Schluß dieses Artikels sei noch etwas zu den unterschiedlichen Sprachversionen von *Fixpacks* gesagt. Einige Leser fragten, ob es möglich ist, auf einem deutschen OS/2-System ein englisches Fixpack zu installieren. Kurz gesagt: Es ist möglich, aber nicht ratsam. Zum einen ist es nicht sonderlich schön, da man eine Mischung aus englischen und deutschen Dialogen nach der Installation eines englischen *Fixpacks* auf einem deutschen System geschaffen hat. Zum anderen können neben dieser eher kosmetischen Auswirkung auch andere Effekte auftreten, die auf Inkompatibilitäten zwischen dem englischen *Fixpack* und der deutschen Betriebssystemversion zurückzuführen sind. Zwar erscheinen die *US-Fixpacks* immer vor den deutschen *FPs*, aber die Abstände sind doch sehr gering, so daß man ruhig ein wenig auf die deutschen Fixpackversionen warten sollte.

Im Anhang finden Sie übrigens einige interessante Webadressen rund um das Thema *Fixpacks*, falls Sie noch weitere Informationen benötigen sollten. Darunter befindet sich auch eine IBM-Adresse, die den Installationsprozeß noch einmal erklärt. Dieses Dokument ist v.a. für all jene zu empfehlen, die mit der Textversion von *service*, *fservice*, arbeiten möchten und Informationen zum Erstellen von Antwortdateien benötigen. Wir haben *fservice* hier nicht betrachtet, da das Programm für den Privatanwender oder den Betreiber kleiner Netzwerke wenig interessant ist.

Zum Abschluß dieses Beitrages sei noch angefügt, daß wir uns mit den anfangs erwähnten *syslevel*-Dateien in einem späteren Artikel noch genauer auseinandersetzen werden, da hier bei vielen Anwendern ein Informationsdefizit besteht. Im Rahmen unserer Programmierworkshops werden wir außerdem zeigen, wie man mit selbstgeschriebenen Programmen Informationen aus diesen Dateien lesen kann. Um Ihnen jetzt schon einen kleinen Überblick zu den *syslevel*-Dateien zu geben, finden Sie im Anhang eine Übersicht zu den wichtigsten Dateien mit ihren jeweiligen Bedeutungen. □

Bootsysteme I

»Never change a running system!« Diesen Satz kennen Sie sicher. Aber man kommt nicht umhin, Veränderungen am System vorzunehmen: Die Installation von Fixpacks z.B. erfordert tiefe Eingriffe in das Betriebssystem. Aber auch die Installation neuer Programme kann OS/2 sehr verändern. So kann es durchaus vorkommen, daß das System nicht mehr so arbeitet, wie man es sich wünschte. Oftmals sind es nur kleine Eingriffe in der *config.sys*, die es wieder zum Laufen bringen. Manchmal ist es aber auch notwendig, die Betriebssystempartition zu formatieren und ein zuvor angefertigtes Backup aufzuspielen. Neben diesen Notfällen gibt es natürlich auch Wartungsarbeiten, die man am System durchzuführen hat, angefangen vom Löschen temporärer Dateien, über das Defragmentieren von Datenträgern bis hin zum Anfertigen der essentiell wichtigen Backups. Wann immer man auch in den »Erste Hilfe-Kasten« für OS/2 greifen muß: Man kann oder sollte das System dazu nicht von der Betriebssystempartition, sondern von Dienstprogrammdisketten starten, damit keine Datei der sonst aktiven Partion durch das System gesperrt, also in Verwendung ist.

Ein weitaus bequemerer Weg ist die Verwendung einer sogenannten **Wartungspartition**. Diese Partition enthält eine abgespeckte Variante des Betriebssystems, die man im Falle eines Falles starten kann, um auf den anderen Laufwerken des Rechners Arbeiten vornehmen zu können. Man kann auf einer solche Partition -im Gegensatz zu Disketten- *alle* wichtigen Werkzeuge installieren und weitaus schneller arbeiten als mit dem Diskettenlaufwerk. Auch bootfähige CDs machen in manchen Fällen Bootdisketten überflüssig. Nichtsdestrotrotz braucht man für wirklich ernste Notfälle, wenn man etwa von keiner Partition der Festplatte mehr starten kann oder über keinen bootfähigen SCSI-oder ATPAI-Adapter verfügt, einige 3,5"-Disketten zum Starten des Rechners.

Wir werden daher in einem zweiteiligen Artikel unterschiedliche Bootsysteme näher betrachten, wie man sie einrichtet und benutzt. Zunächst betrachten wir die Einrichtung und Konfiguration von Wartungspartitionen. Im zweiten Teil dieses Artikels werden wir die Herstellung bootfähiger CDs und von Wartungsdisketten betrachten. Dort werden wir zeigen, wie man alle nötigen Dateien auf nur eine Diskette unterbringt. Vor allem aber zur Herstellung bootfähiger CDs für OS/2 findet man nur sehr spärliche Informationen, so daß wir diesem Thema etwas mehr Raum gewähren.

Der OS/2-Bootprozeß

Bevor wir zur Einrichtung einer Wartungspartition schreiten, wollen wir uns noch mit ein wenig Theorie beschäftigen, um zu erfahren, wie OS/2 überhaupt bootet und welche Dateien dabei eine Rolle spielen. Zur Verdeutlichung des Bootprozesses dient Abb. 1. Wir betrachten hier vorerst nur das Starten des Systems von Diskette und der Festplatte, nicht von einer bootfähigen CD (diesem Prozeß widmen wir uns im zweiten Teil dieses Artikels).

Wird das System gestartet, so liest das **BIOS** (Basic Input Output System) zunächst die **MBRs** (Master Boot Records) aller Festplatten aus. Ein Bootrecord ist ein Bereich auf der Festplatte, der eine startfähige Partition markiert und beschreibt. Festgehalten werden diese *MBRs* übrigens in der sogenannten **Partitionstabelle** (dieser Begriff ist Ihnen sicherlich schon bekannt).

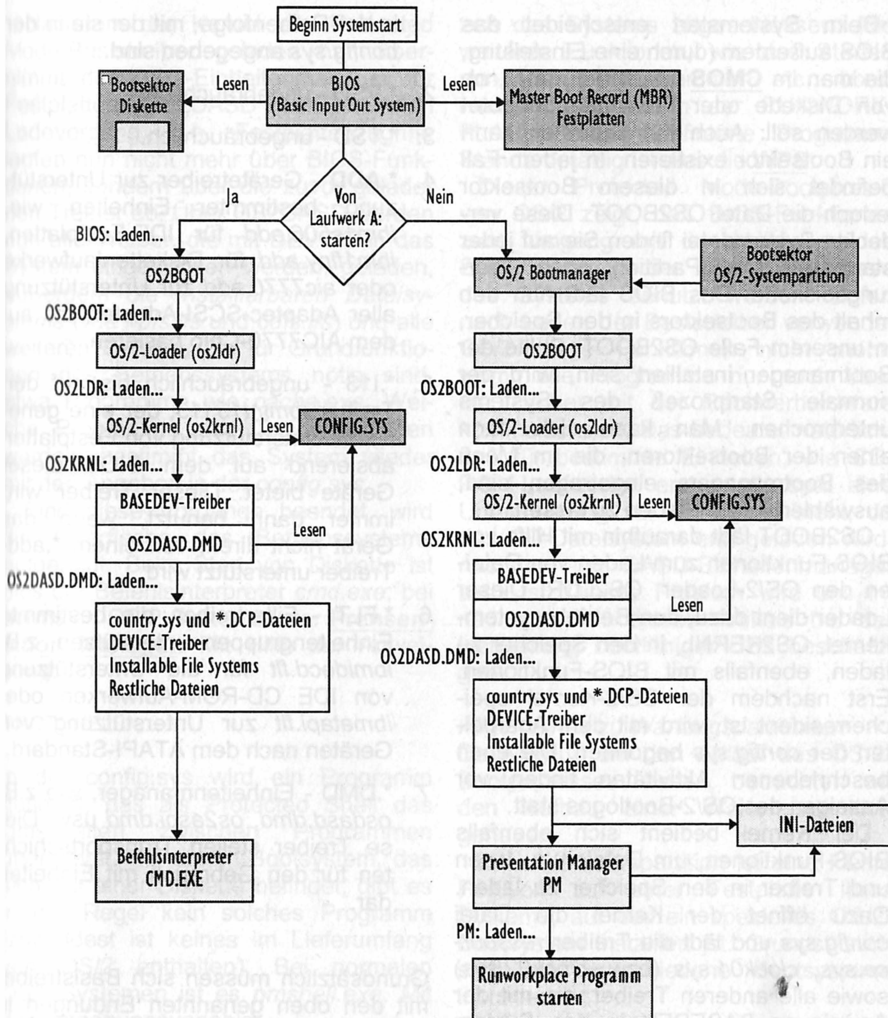


Abb. 1: Der Protected Mode Bootprozeß von OS/2. Zunächst lädt der OS/2-Loader und der Kernel des Betriebssystems die Protected Mode Treiber mit Hilfe von Standard-BIOS-Funktionen. Nach dem Laden aller Treiber, die mit der BASEDEV-Anweisung in das System eingebunden werden, übernimmt der OS/2-Einheitenmanager die Kontrolle des Bootprozesses. Er lädt alle weiteren Treiber und nötigen Dateien nicht mehr mit BIOS-Funktionen, sondern mit Hilfe der zuvor in den Speicher geladenen Protected Mode-Treibern. Mit Hilfe des Bootmanagers wird der Bootsektor der zu startenden Betriebssystempartition ausgesucht, den das BIOS zum Starten des Systems braucht. Die Bootsektoren sind in den MBRs (Master Boot Records) gespeichert und werden vom BIOS abgefragt.

Beim Systemstart entscheidet das BIOS außerdem (durch eine Einstellung, die man im **CMOS** speichern kann), ob von Diskette oder Festplatte gebootet werden soll. Auch auf Disketten kann ein Bootsektor existieren. In jedem Fall befindet sich in diesem Bootsektor jedoch die Datei OS2BOOT. Diese verdeckte Systemdatei finden Sie auf jeder startbaren OS/2-Partition oder -Wartungsdiskette. Das BIOS lädt nun den Inhalt des Bootsektors in den Speicher, in unserem Falle OS2BOOT. Sollte der Bootmanager installiert sein, wird der normale Startprozeß des Systems unterbrochen. Man kann sich dann einen der Bootsektoren, die im Menü des Bootmanagers eingetragen sind, auswählen, um ihn laden zu lassen.

OS2BOOT lädt daraufhin mit Hilfe von BIOS-Funktionen zum Laden von Dateien den OS/2-Loader, OS2LDR. Dieser Loader dient dazu, den Betriebssystemkernel, OS2KERNL, in den Speicher zu laden, ebenfalls mit BIOS-Funktionen. Erst nachdem der OS/2-Kernel speicherresident ist, wird mit dem Abarbeiten der *config.sys* begonnen. Die eben beschriebenen Aktivitäten finden vor Anzeigen des OS/2-Bootlogos statt.

Der Kernel bedient sich ebenfalls BIOS-Funktionen, um Dateien zu lesen und Treiber in den Speicher zu laden. Dazu öffnet der Kernel die Datei *config.sys* und lädt die Treiber *ressource.sys*, *clock01.sys* (bzw. *clock02.sys*) sowie alle anderen Treiber, die mit der Anweisung BASEDEV in das System eingebunden werden. Die Reihenfolge ist dabei abhängig von der Dateinamenerweiterung des Treibers. OS/2 lädt die BASEDEV-Treiber in dieser Reihenfolge:

1. ***.SYS** - Basistreiber wie *clock01.sys*, *ressource.sys*, *screen.sys*, und alle weiteren Treiber mit dieser Endung in der Reihenfolge, mit der sie in der *config.sys* angegeben sind.
2. *.VSD - ungebräuchlich.
3. *.TSD - ungebräuchlich.
4. ***.ADD** - Gerätetreiber zur Unterstützung bestimmter Einheiten wie, *ibm1s506.add* für IDE-Festplatten, *ibm1flpy.add* für Diskettenlaufwerke oder *aic7770.add* zur Unterstützung aller Adaptec-SCSI-Adapter, die auf dem AIC7770-Chip basieren.
5. *.I13 - ungebräuchlich bis auf den Treiber *ibmint13.I13*, der eine generische Unterstützung von Festplatten absierend auf dem BIOS dieser Geräte bietet. Dieser Treiber wird immer dann benutzt, wenn das Gerät nicht direkt von einem *.add-Treiber unterstützt wird.
6. *.FLT - Filtertreiber, die bestimmte Einheitengruppen unterstützen, z.B. *ibmidecd.flt* für die Unterstützung von IDE CD-ROM-Aufwerken oder *ibmatapi.flt* zur Unterstützung von Geräten nach dem ATAPI-Standard.
7. *.DMD - Einheitenmanager, wie z.B. *osdasd.dmd*, *os2aspi.dmd* usw. Diese Treiber stellen Transportschichten für den Gebrauch mit Einheiten dar.

Grundsätzlich müssen sich Basistreiber mit den oben genannten Endungen in einem der folgenden Verzeichnisse befinden:

\ (Stammverzeichnis Bootlaufwerk)
\os2
\os2\boot

OS/2 durchsucht ausschließlich diese Verzeichnisse nach den erforderlichen Dateien.

Nachdem der Kernel alle Protected Mode-Basistreiber geladen hat, übernimmt der OS/2-Einheitenmanager für Festplatten OS2DASD den weiteren Ladevorgang, d.h. Festplattenzugriffe laufen nun nicht mehr über BIOS-Funktionen, sondern über die zuvor geladenen Treiber ab. Über OS2DASD werden nun alle Treiber, die mit DEVICE in das System eingebunden werden, geladen, außerdem die *Installierbaren Dateisysteme* (wie *hpfs.ifs* und *cdfs.ifs*) und alle weiteren Dateien, die für Grundfunktionen des Betriebssystems nötig sind, etwa Programme wie *cache.exe*. Welche Programme und Treiber geladen werden, bestimmt das System wieder mit den Angaben in der *config.sys*.

Sind diese Vorgänge beendet, wird die Oberfläche des Betriebssystems aufgebaut. Beim Start von Diskette ist dies der Befehlsinterpreter *cmd.exe*, bei normalen OS/2-Systemen der **Prensentation Manager**. Mit Hilfe der Anweisung:

PROTSHELL=...

In der config.sys wird ein Programm definiert, das als Protected Shell das Umschalten zwischen Programmen übernimmt. Bei einem Bootsystem, das sich auf einer Diskette befindet, gibt es in der Regel kein solches Programm (zumindest ist keines im Lieferumfang von OS/2 enthalten). Bei normalen OS/2-Systemen ist es *pmshell.exe*. Mit der Umgebungsvariable:

SET RUNWORKPLACE=...

wird bei üblichen OS/2-Bootsystemen die Benutzeroberfläche des Betriebssystems festgelegt. Der *Presentation Manager* überprüft diese Umgebungsvariable und startet das über sie angegebene Programm. Damit ist der Startprozeß des Systems abgeschlossen. Alle weiteren Funktionen, wie das Starten von Benutzerprogrammen etc., übernimmt dann das in der RUNWORKPLACE-Variable definierte Programm. Standardmäßig ist dies die **WPS**.

Dieser Protected Mode-Bootprozeß von OS/2 zeigt, daß BIOS-Funktionen zum Umgang mit der Hardware des Systems nur in der frühen Startphase des Systems benutzt werden. Sofort nach Laden der Basistreiber verwendet OS/2 diese Funktionen nicht mehr, womit alle Programme nur noch über die Treiber mit der Rechnerhardware kommunizieren. Das bedeutet natürlich, daß für bestimmte Einheiten, wie CD-ROM-Laufwerke, erst relativ spät eine Unterstützung zur Verfügung steht, da das CD-Dateisystem erst geladen wird, nachdem der DASD-Einheitenmanager aktiv ist. Dieses Problem wird uns im nächsten Teil dieses Artikels bei der Herstellung bootfähiger CDs beschäftigen.

Allgemeines zu Wartungspartitionen

Da wir nun um den Ablauf des OS/2-Bootprozesses wissen, betrachten wir den Aufbau einer Wartungspartition genauer.

Eine Wartungspartition ist eine kleine Partition auf einer Festplatte Ihres Systems, auf der ein spezielles OS/2-Bootsystem installiert ist. Grundsätzlich sind drei verschiedene Bootsysteme installierbar:

1. *Ein OS/2-System, das im Textmodus läuft.*

   Nach dem Starten wird ein OS/2-Gesamtbildschirm angezeigt. Weitere OS/2-Sitzungen können gestartet werden. Fenstersitzungen oder das Ausführen von PM-Programmen sind nicht möglich.

2. *Ein OS/2-System mit PM-Unterstützung.*

   Der *Prensentation Manager* ist die primäre Anwendungsumgebung des Bootsystems. OS/2-Gesamtbildschirm und -Fenstersitzungen sind möglich, ebenso die Ausführung von PM-Programmen. WPS-Anwendungen können nicht ausgeführt werden.

3. *Ein OS/2-System mit der WPS als Arbeitsumgebung.*

   Zusätzlich zum PM wird die WPS als Arbeitsumgebung installiert. Derartige Bootsysteme sind für Wartungspartitionen wenig empfehlenswert.

Man kann diesen drei Bootsystemtypen nach Bedarf noch unterschiedliche Systemkomponenten hinzufügen, etwa eine DOS- oder Windows-Unterstützung, REXX etc. Wer sich gut genug mit dem System auskennt, kann sich ein speziell auf seine Bedürfnisse zugeschnittenes Bootsystem erzeugen.

Es gibt jedoch schon seit Jahren ein kleines und sehr hilfreiches Werkzeug namens *bootos2*. Dieses Programm kann unterschiedliche Bootsysteme auf Disketten und Festplattenpartitonen erzeugen. Die Parameter, die man dem Programm übergeben kann, sind vielfältig, so daß man die Wartungspartition, welche man benötigt, mit diesem Utility in der Regel erzeugen kann. Wir werden im nächsten Abschnitt *bootos2* vorstellen. Dennoch wird man manchmal nicht umhinkönnen, selbst noch einige Änderungen an der Wartungspartition, die *bootos2* einrichtet, vorzunehmen. Vor allem im nächsten Teil dieses Artikels werden wir uns damit eingehender beschäftigen. Aber auch an einer normalen Wartungspartition sind z.T. Änderungen nötig. Auf Veränderungen, die man manuell durchführen sollte, gehen wir ebenfalls ein.

Bevor wir uns mit *bootos2* beschäftigen ist zu klären, welche Vorbereitungen Sie zu treffen haben. Sie benötigen zum Einrichten einer Wartungspartition natürlich erst einmal einen kleinen Bereich auf der Festplatte, der die Dateien des Bootsystems aufnimmt. Abhängig von der Art des Bootsystems müssen Sie die Größe der erforderlichen Partition festlegen. Für ein normales Wartungssystem, das nur die Arbeit mit OS/2- und/ oder DOS-Anwendungen in Gesamtbildschirmsitzungen erlaubt, geben Sie der Partition mindestens 3 MByte Festspeicher, jedoch nicht mehr als 6 MByte. Möchten Sie lieber mit dem PM als Anwendungsumgebung arbeiten, lassen Sie die Partition mindestens 12 MByte, jedoch nicht mehr als 16 MByte groß werden. Von der Verwendung der WPS oder der Einrichtung einer Windows-Unterstützung für kleine Bootsysteme raten wir ab. Derartige Systeme sind schon keine Wartungssysteme mehr und brauchen zu viel Speicherplatz. Auch eine PM-Wartungspartition ist keine optimale Lösung, da auch hier die Speicherplatzerfordernisse viel höher liegen als bei einem System, das nur im Textmodus arbeitet. Beachten Sie im allgemeinen:

- Wartungssysteme sollten schnell zu starten sein und auch so schnell wie möglich laufen,
- wenig Platz auf der Festplatte wegnehmen,
- nur dann eine DOS-Unterstützung bieten, wenn auch mit FAT formatierte Partitionen gewartet werden sollen oder wenn bestimmte Hilfsprogramme, mit denen Sie

arbeiten, nur für DOS zur Verfügung stehen und

- zusätzliche aber nicht unbedingt nötige Systemkomponenten und Programme nur dann aufnehmen, wenn sie tatsächlich benötigt werden.

Alles andere ist, auch wenn man es nicht sehr gerne hören mag, Spielerei. Ein Wartungssystem muß nicht schön und komfortabel sein: Es muß nur funktionieren. Der Ratschlag, das System so einfach wie nur irgend möglich zu halten, hat einen ganz pragmatischen Grund: Je komplizierter ein System ist, desto größer ist seine Anfälligkeit gegenüber Störungen. Wir verwenden auf einem unserer Rechner ein kleines Bootsystem für Wartungszwecke, das mit Erscheinen von Warp 3 eingerichtet wurde und immer noch läuft, ohne das jemals ein Problem aufgetreten oder eine Änderung nötig geworden wäre. Und selbst wenn Sie einmal Schwierigkeiten mit dem Wartungssystem haben sollten: Ist es einfach aufgebaut, ist das Problem meistens sehr schnell behoben. Wenn man noch ein Wartungssystem für das Wartungssystem braucht, dann hat man sein Ziel verfehlt. Wir werden daher Wartungssysteme nach den oben angegebenen Richtlinien erzeugen.

Installation von bootos2

Das Programm *bootos2* liegt mittlerweile in der Version 9.15 vor. Sie erhalten es als kleine ZIP-Datei von allen OS/2-Software-Servern und auch auf unserer Homepage im Downloadbereich der *OS/2 Only!*-Seiten. Besorgen Sie sich außerdem gleich noch *LxLite*. Dieses Programm komprimiert ausführbare Programmdateien und ist bei der Erzeugung von Bootsystemen immer zu empfehlen. Manchmal sind es nur einige Byte, die auf einer Diskette fehlen, und die man durch eine bessere Komprimierung der ausführbaren Programmdateien wieder verfügbar machen kann. *LxLite* in der Version 1.21 haben wir ebenfalls auf unserer Homepage für Sie bereitgelegt. Die Installation von *bootos2* und *LxLite* ist einfach:

❶ Erzeugen Sie ein Verzeichnis namens *bootos2* und entpacken Sie darin das Archiv *btos2915.zip*.

❷ Fügen Sie das Installationsverzeichnis von *bootos2* der PATH-Anweisung in der *config.sys* hinzu, um das Programm von jedem Verzeichnis aus starten zu können.

Zur Installation von *LxLite* verfahren Sie wie folgt:

❶ Entpacken Sie die Datei *lxlt121.zip* in einem temprären Verzeichnis. Die weitere Installation erfolgt mit einem mitgelieferten Installationsprogramm.

❷ Öffnen Sie ein OS/2-Fenster, wechseln Sie in das Verzeichnis, in dem Sie das *LxLite*-Archiv entpackt haben, dort in das beim Entpacken erstellte Verzeichnis *lxlite*, und geben Sie hier den Befehl *install* ein.

❸ Das Installationsprogramm fragt Sie nach dem Verzeichnis, in dem *LxLite* installiert werden soll. Geben Sie hier das Zielverzeichnis an, z.B.:

d:\programme\lxlite121

Existiert das Verzeichnis nicht, wird es erstellt.

Die wichtigsten Parameter von bootos2

SOURCE=<Verzeichnis>
Bezeichnet das Verzeichnis, in dem sich alle Systemdateien befinden, die *bootos2* für den Installationsprozeß des Bootsystems benötigt (Standard: *Bootos2* fragt nach der Installationsdikette). Das Programm benötigt die Datei *sysinstx.com* zur Erzeugung des OS/2-Bootrecords; wird das Bootsystem auf einer Diskette erstellt, sind außerdem die beiden Dateien *keyboard.dcp* und *vtbl850.dcp* sowie die Datei *config.sys* der Installationsdiskette nötig. Möchte man diese Diskette für die Einrichtung des Wartungssystems nicht verwenden, gibt man mit SOURCE ein Verzeichnis an, in dem sich die genannten Dateien befinden. Beispiel: *bootos2 source=h:\bootfiles target=e:*

TARGET=<LW>
Bezeichnet das Laufwerk, auf dem das Bootsystem installiert werden soll (Standard: A). Das Laufwerk wird ohne Doppelpunkt angegeben. *Beispiel: bootos2 target=f*

TYPE=<PM | WPS>
Gibt den Typ des zu erzeugenden Bootsystems an. (Standard: Textmodus-Bootsystem). *PM* steht für die Erzeugung eines PM-Bootsystems; bei *WPS* wird zusätzlich zur PM- die WPS-Unterstützung installiert. Dieser Parameter kann nicht zusammen mit TARGET=A angegeben werden.

2DISK=<LW>
Dieser Parameter bewirkt, daß Bootsysteme, die auf Diskette installiert werden, auf zwei Disketten aufgeteilt werden. Wenn Sie nur ein Diskettenlaufwerk haben, benötigt der Parameter das Argument <LW> nicht. Haben Sie zwei Diskettenlaufwerke, benutzt *bootos2* standardmäßig Laufwerk B: als Ziellaufwerk für die zweite Diskette. In diesem Falle können Sie <LW> auf A: setzen, damit *bootos2* Laufwerk B: nicht zur Erzeugung der 2. Bootdiskette benutzt. *Beipsiel: bootos2 2disk=a*

REXX
Durch Angabe dieses Parameters können Sie auf dem Wartungssystem REXX installieren lassen, sollte genügend Platz zur Verfügung stehen. Für alle Bootsysteme möglich (auch Diskettensysteme).

HELP
Mit diesem Parameter können Sie PM- und WPS-Bootsystemen den OS/2-Helpmanager hinzufügen. Nur eine Basisunterstützung des Online-Hilfesystems wird installiert (nur der Helpmanager mit den wichtigsten Hilfedateien).

VDM
Das Bootsystem wird durch eine DOS-Unterstützung erweitert. Sie können diesen Parameter bei allen Bootsystemen angeben (auch Disketten, sofern Sie mindestens Warp 3 einsetzen). Auf einem Textmodussystem starten Sie eine neue DOS-Sitzung durch die Tastenkombination Strg+D.

VGA
Dieser Parameter wird nur für PM- und WPS-Bootsysteme benötigt. Er bewirkt, daß auf dem Bootsystem die Standard-VGA-Graphiktreiber installiert werden. Standardmäßig installiert *bootos2* die Graphiktreiber des während der Einrichtung der Wartungspartition aktiven OS/2-Systems.

Die wichtigsten Parameter von bootos2 (Fortsetzung)

LX
Wird dieser Parameter angegeben, komprimiert *bootos2* alle Dateien mit den Erweiterungen *.exe, *.com, *.dll, *.sys, *.ifs, *.add, *.flt und *.dmd sowie den Betriebssystemkernel os2krnl. *Bootos2* benötigt dazu das Freewareprogramm *LxLite* (mindestens ab Version 1.2.1). Die Dateien *lxlite.cfg* und *lxlite.exe* müssen sich im gleichen Verzeichnis wie *bootos2.exe* befinden (zur Installation siehe Text: Installation von *bootos2*).

SYSED
Mit diesem Parameter fügt *bootos2* die Dateien des Systemeditors dem Bootsystem hinzu. Der Parameter kann nur für PM- bzw. WPS-Bootsysteme angegeben werden.

SWAP=<Verzeichnispfad>
Gibt das Verzeichnis der Datei *swapper.dat* an (Standard: Stammverzeichnis des Laufwerkes, auf dem das Bootsystem installiert wird). Der Parameter ist nur für PM- und WPS-Bootsysteme gültig. Beispiel: *bootos2 target=d type=pm swap=d:\os2\swap*

FILE=<Textdatei>
Mit dem Parameter FILE wird *bootos2* der vollständige Dateiname einer Textdatei übergeben. Die in dieser Datei angegebenen Dateien werden von *bootos2* zusätzlich zu den standardmäßig zur Installation des Bootsystems benötigten Dateien auf das Laufwerk des zu erzeugenden Bootsystems in das Verzeichnis *\os2* kopiert. Die Datei muß eine reine ASCII-Datei sein. Leere Zeilen in dieser Datei oder solche, die mit einem Stern (*) beginnen, werden von *bootos2* ignoriert. Beginnt eine Zeile mit einem Gleichheitszeichen (=), betrachtet *bootos2* den dahinter folgenden Text als Pfadangabe für eine Datei. Hier muß der vollständige Dateiname angegeben werden, also mit Laufwerk und Verzeichnispfad. (Weitere Informationen zu FILE mit Beispielen im Text.)

TRACE=<Zieldatei>
Wird dieser Parameter angegeben, erzeugt *bootos2* eine Datei, in dem der Installationsprozeß des Bootsystems dokumentiert wird. Diese Datei heißt *bootos2.log* und wird standardmäßig im Programmverzeichnis von *bootos2* abgelegt, wenn *<Zieldatei>* nicht angegeben wird. Wünscht man, daß *bootos2* die Protokolldatei in einem anderen Verzeichnis unter anderem Namen ablegt, muß der vollständige Name der Protokolldatei angegeben werden. Beispiel: *bootos2 trace=d:\os2\system\bootsys.log*

FORMAT:<FAT | HPFS | NONE>
Durch Angabe dieses Parameters formatiert *bootos2* das mir TARGET angegebene Laufwerk für das durch das Argument angegebene Dateisystem. Mit *NONE* wird eine Formatierung verhindert (Standard: *FAT*). Wird dieser Parameter nicht angegeben, fragt *bootos2* vor dem Formatieren nach dem zu benutzenden Dateisystemtyp. Für Wartungssysteme, die auf einer Festplatte installiert werden, geben Sie auch bei kleinen Partitionsgrößen stets *HPFS* an.

QUIET
Geben Sie diesen Parameter an, erzeugt *bootos2* keinen Output (weder auf dem Bildschirm, noch über den Lautsprecher). Beachten Sie hierzu: Geben Sie immer den Parameter FORMAT an, wenn QUIET angegeben wird, da *bootos2* das durch TARGET angegebene Ziellaufwerk standardmäßig mit FAT-formatiert. Geben Sie außerdem bei Bedarf den Parameter TRACE an, um eine Protokolldatei erzeugen zu lassen.

❹ Das Programm fragt anschließend, welche Teile von *LxLite* installiert werden sollen. Bestätigen Sie die folgenden Fragen wie folgt:

Install lxLite utility pack? (Y/N) **Y**
Install lxLite text resources? (Y/N) **N**
Install lxLite documentation (Y/N) **Y**
Create lxLite folder? (Y/N) **Y**

Anschließend werden die Dateien in das zuvor angegebene Installationsverzeichnis kopiert und der *LxLite*-Ordner auf der Arbeitsoberfläche erstellt.

❺ Beantworten Sie die folgenden Fragen des Installationsprogrammes:

Installation complete. Do you want to read the WHATSNEW file now? **N**

Do you wish to clean the install source directory? **N**

Das Installationsprogramm konnte die Verzeichnisse des Quellverzeichnisses in unseren Tests nicht vollständig löschen. Geben Sie daher *N* ein, und nehmen Sie das Beseitigen der entpackten Archivdateien selbst vor.

❻ Wechseln Sie in das Installationsverzeichnis von *LxLite* und kopieren Sie die Dateien:

lxlite.exe
lxlite.cfg

in das Installationsverzeichnis von *bootos2*.

❼ Führen Sie einen Systemabschluß durch, und starten Sie das System neu, um die Änderungen wirksam werden zu lassen.

Damit ist die Installation der beiden Programme abgeschlossen. *LxLite* ist übrigens nicht nur ein Komprimierungsprogramm für Programmodule; das kleine Softwarepaket verfügt auch noch über andere interessante Werkzeuge. Leider können wir im Rahmen dieses Artikels nicht detailliert auf *LxLite* eingehen. Wir werden uns jedoch in einer späteren Ausgabe mit dem Programm eingehend beschäftigen.

Wir richten eine Wartungspartition ein

Wir beschäftigen uns nun mit der Einrichtung der Wartungspartition mit *bootos2*. Dazu müssen wir zunächst eine geeignete Partition erstellen. Da wir ein ausgesprochen kleines Wartungssystem erzeugen, benötigen wir eine Partition von nur etwa 6 Mbyte Größe.

Sollte man vor der Entscheidung stehen, OS/2 neu zu installieren, oder richtet man häufig neue Systeme ein, so sollte man bereits vor der Installation ein wenig Platz für eine Wartungspartition bereitstellen. Am besten ist es, wenn sich diese Partition auf einem anderen physikalischen Laufwerk befindet als das später installierte OS/2-System. Jedoch ist bereits eine Wartungspartition auf dem gleichen physikalischen Laufwerk eine gute Wahl. Man erzeugt die Wartungspartition mit *fdisk* als logisches Laufwerk in einer erweitertern Partition und trägt dieses am besten unter dem Namen *Wartung* in das Menü des Bootmanagers ein.

! ***Installieren Sie den Bootmanager immer, auch wenn Sie nur OS/2 auf Ihrem System einsetzen. Sie können sonst später von keiner anderen Partition starten, was für ein Bootsystem für Wartungszwecke notwendig ist.***

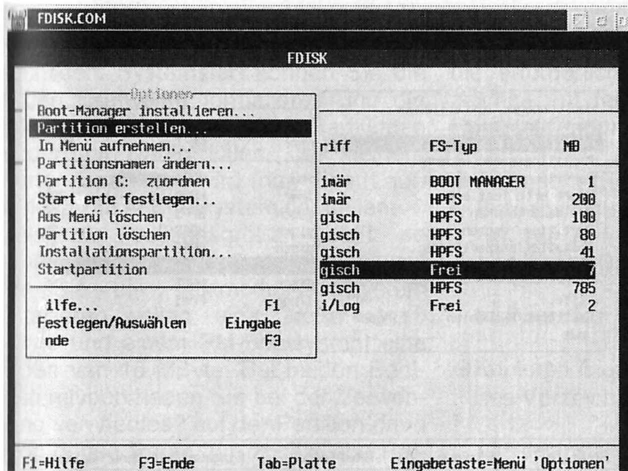


Abb. 2: Eine Partition mit FDISK erstellen.

Wenn Ihr System bereits komplett eingerichtet ist, und Sie verfügen über keine Partition, die das Wartungsbootsystem aufnimmt, ist das Erzeugen unserer Partition etwas schwieriger. Wer ein Tool wie *PartitionMagic* hat, kann die Größe der vorhandenen Partitionen so anpassen, daß Platz für das Wartungssystem bereitsteht. Nennt man ein derartiges Werkzeug nicht sein eigen, kommt man nicht umhin, den Inhalt einer Partition zu sichern, diese zu löschen und den entstandenen Platz entsprechend neu aufzuteilen. Es empfiehlt sich bei einer solchen Vorgehensweise, stets die Partition zu sichern und zu löschen, die am Ende der Laufwerksliste steht. Verfügt man z.B. über drei logische Laufwerke, C:, D:, und E:, sichert die Partition D:, löscht und partitioniert sie neu, so daß man ein zusätzliches, kleines logisches Laufwerk zur Verfügung hat, kann das neue Laufwerk z.B. den Buchstaben D: haben. Damit wird das ehemalige Laufwerk D: zu Laufwerk E: und E: zu F:, was dazu führt, daß auf E: bzw. D: installierte Programme bei einem späteren Systemstart nicht mehr gefunden werden können. Besser ist es also, in unserem Beispiel das Laufwerk E: für eine Neupartitionierung entsprechend vorzubereiten. Die *Abbildungen 2-4* zeigen, wie man eine Partition mit *fdisk* erstellt.

Sie können *fdisk* von einem OS/2-Fenster aus starten, sind also nicht gezwungen, das System von Diskette zu booten, um dann *fdisk* auszuführen.

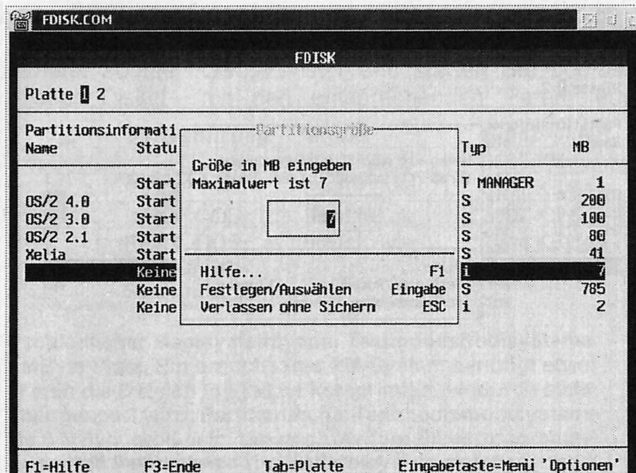


Abb. 3: Größe der zu erstellenden Partition angeben.

Nachdem Sie wissen, welches Laufwerk neu partitioniert werden soll, sichern Sie zunächst dessen Inhalt. Anschließend öffnen Sie ein OS/2-Fenster und starten *fdisk*. Das Programm zeigt alle Laufwerke Ihres Systems und die Partitionen darauf an. Wählen Sie das in Frage kommende Laufwerk aus und drücken Sie *Enter*, um das Menü *Optionen* anzeigen zu lassen (*Abb. 2*). Dort wählen Sie den Menüpunkt *Partition erstellen*, worauf der Dialog *Partitionsgröße* angezeigt wird (*Abb. 3*). Standardmäßig ist die der maximal auf der Partition verfügbare Speicherplatz eingetragen. Ziehen Sie hier die Größe der Wartungspartition ab, und erstellen Sie das Laufwerk, welches die Daten aufnimmt, die auf der gelöschten Partition lagen. Anschließend selektieren Sie den von *fdisk* angezeigten freien Speicherbereich und drücken *Enter* zum Anzeigen des *Optionsmenüs* (*Abb. 2*). Wählen Sie wieder den Menüpunkt *Partion erstellen* aus. Weisen Sie den gesamten verbleibenden Speicherplatz der kleinen Partition zu, die am Ende der Laufwerksliste stehen sollte. Nachdem die Partition erstellt wurde, rufen Sie nochmals das Menü *Optionen* auf und wählen dort den Menüpunkt *In Menü aufnehmen* (*Abb. 4*). Im dem daraufhin erscheinenden Dialog *Neuer Name* (*Abb. 5*) geben Sie die Bezeichnung der Partition ein, die im Menü des Bootmanagers wieder auftauchen wird. Nennen Sie die Partition am besten *Wartung*. Beenden Sie *fdisk* durch Drücken der

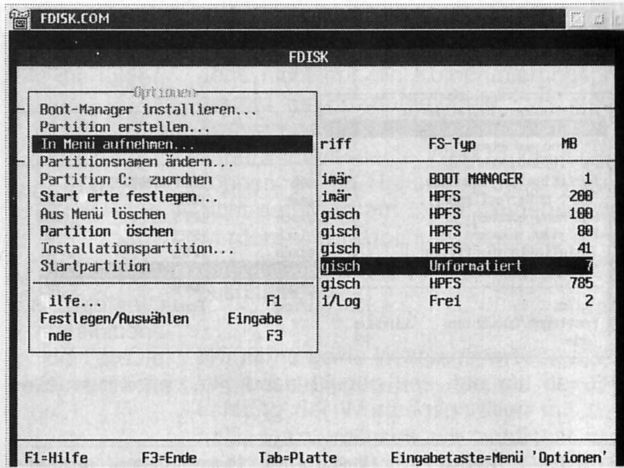


Abb. 4: Die Wartungspartition in das Bootmenü aufnehmen

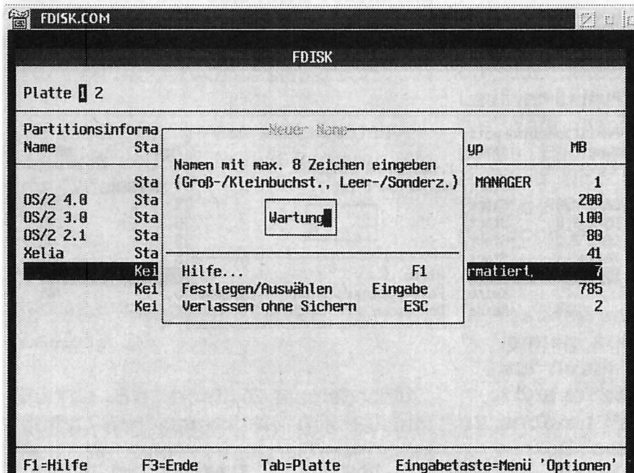


Abb. 5: Den Namen der Partition eingeben

Taste *F3*, um die Änderungen zu speichern, und starten Sie Ihr System neu. Nach dem Systemstart können Sie die beiden Laufwerke formatieren und die größere der beiden neuen Partitionen wieder mit Daten füllen. Die kleinere Wartungspartition wird nun das Laufwerk für unser Bootsystem darstellen.

Was die Partitionsgrößen betrifft, so geben Sie einer Wartungspartion, auf der Sie ein Textmodus-Bootsystem installieren wollen, eine Größe von 6 MByte und einem PM-Bootsystem eine Größe von 16 MByte. Bei beiden Bootsystemtypen haben Sie bei der Verwendung von *bootos2* auf der Partition dann noch genug Raum, um zusätzliche Programme zu installieren. Die genauen Größen der einzelnen Bootsysteme entnehmen Sie bitte der *Tabelle 1*.

Das Bootsystem wird installiert

Nach dem Einrichten der Wartungspartition, können wir mit *bootos2* das Bootsystem auf dieser Partition installieren. Sie benötigen dazu einige Dateien von den Installationsdisketten: *sysinstx.com* zur Erstellung des OS/2-Bootrecords, die **.dcp*-Dateien und die *config.sys*, die auf der 1. Installationsdiskette gespeichert ist. Ohne den Parameter *source* fragt Sie *bootos2* nach diesen Disketten. Damit man sich diese nicht immer bereitlegen muß, kopieren Sie einfach die erforderlichen Dateien in ein Verzeichnis auf der Festplatte, das Sie dem Parameter *source* als Wert übergeben:

➊ Erzeugen Sie am besten auf Ihrer Betriebssystempartition ein Verzeichnis zur Aufnahme der von *bootos2* benötigten Dateien, z.B.:

\os2\bootfiles

➋ Kopieren Sie folgende Dateien in dieses Verzeichnis:

Von der Installationsdiskette:
sysinstx.com

Von Diskette 1:
config.sys
(Bei Warp 3)
keyboard.dcp
vtbl850.dcp

(Bei Warp 4)
Von Diskette 2:
keyboard.dcp
vtbl850.dcp

Anschließend können Sie das Bootsystem installieren. Öffnen Sie dazu ein OS/2-Fenster und starten Sie *bootos2* mit den entsprechenden Parametern.

Typ	***Modus***	***Unterstützung***	***Größe***	***Komprimiert***
Minimal I	Text	--------	3282 KByte	3003 KByte
Minimal II	Text	REXX	3585 KByte	3207 KByte
Minimal III	Text	REXX, DOS	3990 KByte	3607 KByte
PM	Graphik	REXX, DOS, HelpManager, Systemeditor	8628 KByte	7844 KByte

Tabelle 1: Platzbedarf unterschiedlicher Bootsystemtypen. Textmodusbootsysteme brauchen ca. 3,5 MByte Platz. Ein brauchbares PM-System benötigt etwa 7,6 MByte, sofern man die Dateien mit LxLite komprimiert, wodurch stets 9-10 % Festspeicher gespart wird. Partitionen für Textmodusbootsysteme sollten daher 5 bis 6 MByte groß sein, um noch weitere Dienstprogramme speichern zu können. Die Partition für ein PM-Bootsystem sollte man um ca. 2 MByte für weitere Programme und 5 MByte für die Swap-Datei vergrößern, womit sich Partitionsgrößen von 14-16 MByte ergeben.

Damit Sie einige Beispiele zur Verfügung haben, erzeugen wir die in der Tabelle angegebenen Bootsysteme. Dabei nehmen wir an, die Betriebssystempartition sei Laufwerk G: und die von den Installationsdisketten kopierten Dateien befinden sich im Verzeichnis *\os2\bootfiles* auf Laufwerk C:, das unsere Betriebssystempartition darstellt. Geben Sie folgende Befehle ein, um folgende Bootsysteme zu erzeugen:

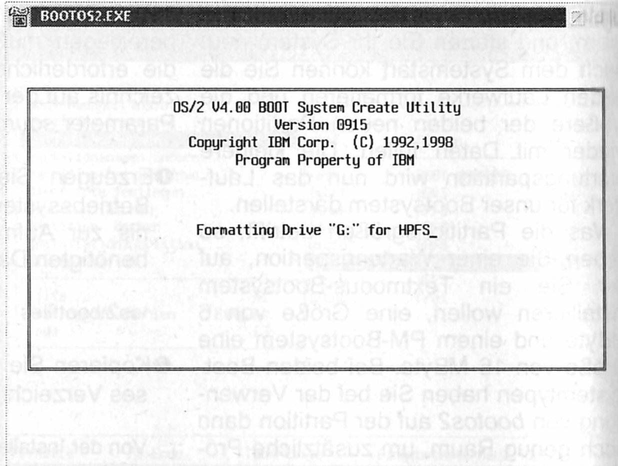


Abb. 6: bootos2 formatiert die Wartungspartition

Für Minimal I:
bootos2 source=c:\os2\bootfiles target=g format:hpfs lx trace

Für Minimal II:
bootos2 source=c:\os2\bootfiles target=g format:hpfs lx rexx trace

Für Minimal III:
bootos2 source=c:\os2\bootfiles target=g format:hpfs lx rexx vdm trace

Für PM:
bootos2 source=c:\os2\bootfiles target=g format:hpfs lx type=pm help sysed vga rexx vdm trace

Mit dem Parameter *trace* erzeugen wir in allen Fällen eine Protokolldatei. Man sollte sich diese Datei nach der Installation des Bootsystems ruhig einmal ansehen, damit man weiß, was *bootos2* gemacht hat. Durch die Angabe von *lx* können wir die meisten Dateien komprimieren lassen. Die Platzersparnis von ca. 10 % sollte man sich nicht entgehen lassen. Die Bedeutung aller anderen Parameter können Sie im Kasten *Die wichtigsten Parameter von bootos2* bei Bedarf noch einmal nachlesen. Erwähnung verdient noch der Parameter *vga*, mit dem man auf dem PM-Bootsystem die Standard-VGA-Treiber installiert. Wir tun das aus drei Gründen und empfehlen daher die Angabe des Parameters:

1. VGA ist für eine Wartungspartition und die darauf installierten Programme völlig ausreichend.

2. Es können beim späteren Start des Bootsystems Schwierigkeiten mit den auf Ihrem OS/2-System eingesetzten SVGA-Treibern geben, die *bootos2* auf die Wartungspartition kopiert hat. In unseren Tests ließ sich das Wartungssystem teilweise nicht mehr starten.

3. Wenn Sie Ihre Graphikkarte wechseln und neue Treiber installieren, müssen Sie dies auch für das kleine OS/2-System auf der Wartungspartition tun. Erfahrungsgemäß vergißt man so etwas, da man die Notpartition nicht jeden Tag benutzt. Sie halten ein PM-System immer funktions-

fähig, wenn es nur mit den VGA-Treibern arbeitet, egal welche Graphikkarte Sie einsetzen.

Nachdem *bootos2* gestartet wurde, wird das Ziellaufwerk zunächst formatiert, in unserem Falle mit HPFS (*Abb. 6*). Formatieren Sie Ihre Wartungspartitionen immer mit HPFS, auch wenn die Partitionen sehr klein sind. Sie genießen so alle Vorteile dieses Dateisystems, angefangen von langen Dateinamen bis hin zu EAs. Um das Laufwerk zu formatieren, benutzt *bootos2* übrigens das Programm *format*. Anschließend legt *bootos2* mit *sysinstx.com* den OS/2-Bootrecord an (*Abb. 7*). *Bootos2* ruft *sysinstx.com* zum Anlegen des Bootrecords wie folgt auf:

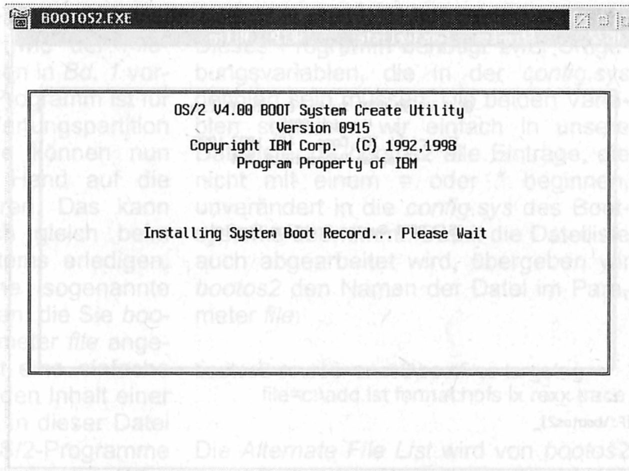


Abb. 7: Der OS/2-Bootrecord (OS2BOOT) wird installiert.

sysinstx <lw>

Dabei ist *<lw>* das im Parameter *target* angegebene Ziellaufwerk. Im nächsten Teil des Artikels werden wir selbst Bootdisketten und -Images für CDs mit diesem Programm erstellen.

Schließlich werden alle erforderlichen Dateien auf die Wartungspartition kopiert *(Abb. 8)*. Während des Kopiervorgangs komprimiert *bootos2* die Dateien mit *LxLite*, sofern eine Komprimierung möglich ist. Ist der Kopiervorgang abgeschlossen, meldet dies *bootos2* mit einem Bildschirm, wie ihn *Abb. 9* zeigt.

Damit ist die Installation des Bootsystems abgeschlossen. Die Installation der REXX-Unterstützung wird im übrigen extra angezeigt.

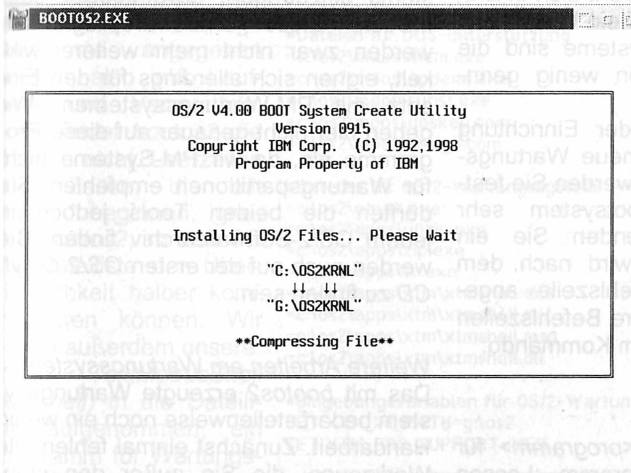


Abb. 8: Alle erforderlichen Dateien werden kopiert.

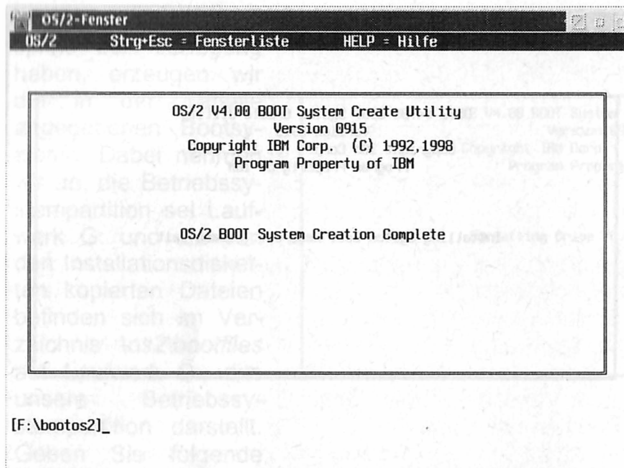


Abb. 9: Der Abschluß der Installation des Bootsystems.

Schwierigkeiten während der Installation können eigentlich nur dann auftreten, wenn auf dem Ziellaufwerk kein Speicherplatz mehr zur Verfügung steht. Wenn Sie sich jedoch an die oben gemachten Angaben halten, ist auf den Wartungspartitionen noch ein ausreichend großer freier Speicherbereich vorhanden. Der Festspeicherbedarf für Wartungspartitionen bezieht sich in diesem Artikel übrigens auf ein Warp 4 mit FP 10. Für Warp 3-Systeme sind die Anforderungen noch ein wenig geringer.

Probieren Sie nach der Einrichtung des Bootsystems Ihre neue Wartungspartition einmal aus, so werden Sie feststellen, daß das Bootsystem sehr schnell startet. Verwenden Sie ein Textmodussystem, so wird nach dem Booten eine OS/2-Befehlszeile angezeigt. Sie können weitere Befehlszeilen oder Programme mit dem Kommando:

start <programm>

aufrufen. Dabei steht *<programm>* für das aufzurufende Programm. Lassen Sie den Parameter fort, wird einfach eine neue Befehlszeile gestartet. Zwischen den einzelnen Sitzungen können Sie mit der Tastenkombination *Alt+Esc* umschalten. Haben Sie eine DOS-Unterstützung installiert, können Sie eine eine DOS-Sitzung mit der Tastenkombination *Strg+V* starten. Zwischen OS/2- und DOS-Sitzungen läßt sich bequem hin- und herschalten, wie Sie es von OS/2 gewohnt sind. Möchten Sie das System neu starten, können Sie dies mit der Tastenkombination *Strg+Alt+Entf* gefahrlos tun.

Ein PM-Wartungssystem präsentiert sich mit einem OS/2-Fenster. Jede Art von Anwendung können Sie hier ebenfalls über den Prompt starten. Wir empfehlen Ihnen jedoch, kleine Shells statt *cmd.exe* in die Variable RUNWORKPLACE der *config.sys* des Wartungssystems einzutragen, etwa *FileBar* oder *MDesk*. Diese Sharewareprogramme werden zwar nicht mehr weiterentwikkelt, eignen sich allerdings für den Einsatz mit PM-Wartungssystemen. Wir gehen hier nicht genauer auf diese Programme ein, da wir PM-Systeme nicht für Wartungspartitionen empfehlen. Sie dürften die beiden Tools jedoch in jedem OS/2-Softwarearchiv finden. Sie werden auch auf der ersten *OS/2 Only!-CD* zu finden sein.

Weitere Arbeiten am Wartungssystem

Das mit *bootos2* erzeugte Wartungssystem bedarf stellenweise noch ein wenig Handarbeit. Zunächst einmal fehlen alle Werkzeuge, die Sie außer den standardmäßig in OS/2 enthaltenen Pro-

grammen einsetzen, etwa ein *unzip.exe* oder ein Dateimanager wie der *File-Commander*, den wir Ihnen in *Bd. 1* vorstellten. Gerade dieses Programm ist für den Einsatz auf einer Wartungspartition vorzüglich geeignet. Sie können nun diese Programme von Hand auf die Wartungspartition kopieren. Das kann *bootos2* allerdings auch gleich beim Einrichten des Bootsystems erledigen. Dazu müssen Sie eine sogenannte *Alternate File List* erstellen, die Sie *bootos2* beim Start im Parameter *file* angeben. Diese Dateiliste ist eine einfache Textdatei. *Abb. 10* zeigt den Inhalt einer Beispieldatei. Wir haben in dieser Datei einige DOS- und OS/2-Programme angegeben, die wir auf unserer Wartungspartition zusätzlich installiert sehen möchten. Neben dem ein oder anderen Dienstprogramm, das mit OS/2 ausgeliefert wird, findet man auch einige Programme aus dem \os2\apps-Verzeichnis, die Sie ebenfalls kennen dürften, etwa *fc.exe* oder *unzip.exe*. Wie Sie sehen, werden Dateien mit einem = in die Dateiliste eingebunden. Hier muß der vollständige Dateiname angegeben werden, also inkl. Laufwerk und Verzeichnis, sonst findet *bootos2* die Dateien nicht. Leerzeilen oder Zeilen, die mit einem * beginnen, ignoriert *bootos2*, so daß Sie die Dateiliste der Übersichtlichkeit halber kommentieren können. Wir haben außerdem unsere *Xelia TextModusShell (xtmshell)* in die Dateiliste aufgenommen, ein Programm für Wartungspartitionen, das sich allerdings noch im Testzustand befindet. Dieses Programm benötigt zwei Umgebungsvariablen, die in der *config.sys* definiert sein müssen. Die beiden Variablen schreiben wir einfach in unsere Dateiliste, da *bootos2* alle Einträge, die nicht mit einem = oder * beginnen, unverändert in die *config.sys* des Bootsystems übernimmt. Damit die Dateiliste auch abgearbeitet wird, übergeben wir *bootos2* den Namen der Datei im Parameter *file*:

```
bootos2 source=c:\os2\bootfiles target=g
        file=c:\add.lst format:hpfs lx rexx trace
```

Die *Alternate File List* wird von *bootos2* zum Ende des Installationsprozesses abgearbeitet. Die entsprechende Statusmeldung zeigt *Abb. 11*.

Auf diese Weise ist Ihr Bootsystem individuell erweiterbar. Nachteilig an dieser an sich schönen Funktion ist vielleicht, das alle Dateien im *\os2*-Ver-

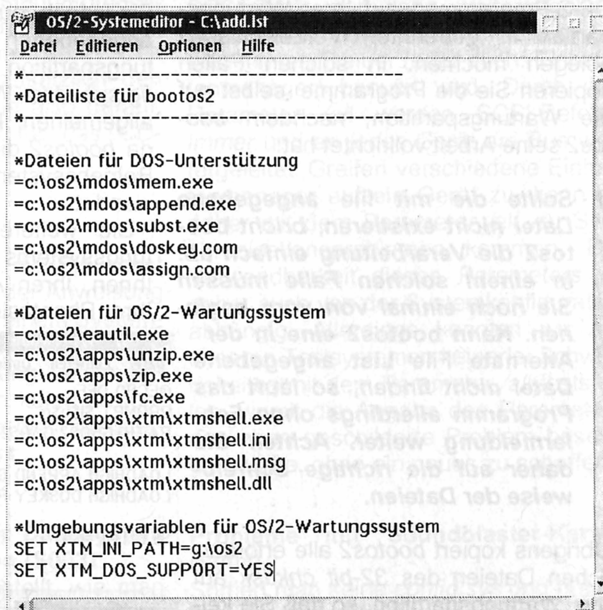


Abb. 10: Beispiel für eine Alternate File List

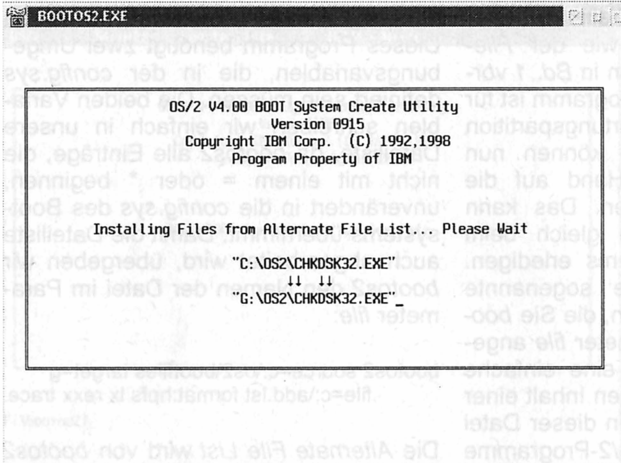


Abb. 11: Die Alternate File List wird abgearbeitet

zeichnis der Wartungspartition abgelegt werden. In der Regel sollte das nicht sonderlich schlimm sein, jedoch kann es in manchen Fällen zu Unübersichtlichkeit führen, gerade wenn Sie DOS von OS/2-Programmen auf Ihrer Wartungspartion in getrennten Verzeichnissen ablegen möchten. In solchen Fällen kopieren Sie die Programme selbst auf die Wartungspartition, nachdem *bootos2* seine Arbeit vollrichtet hat.

! ***Sollte die mit* file *angegebene Datei nicht existieren, bricht* bootos2 *die Verarbeitung einfach ab. In einem solchen Falle müssen Sie noch einmal von vorn beginnen. Kann* bootos2 *eine in der* Alternate File List *angegebene Datei nicht finden, so läuft das Programm allerdings ohne Fehlermeldung weiter. Achten Sie daher auf die richtige Schreibweise der Dateien.***

Übrigens kopiert *bootos2* alle erforderlichen Dateien des *32-bit chkdsk* auf Ihre Wartungspartition, so daß Sie keine entsprechenden Einträge in der Dateiliste erstellen müssen. Haben Sie auf Ihrer Wartungspartition eine DOS-Unterstützung installiert, sollten Sie aber noch eine *autoexec.bat* anlegen, da *bootos2* dies nicht tut. Im wesentlichen dient diese *autoexec.bat* der Konfiguration des **Prompts**. Daher sind in unserem Beispiel für eine *Alternate File List* auch *append.com* und *doskey.com* angegeben. Um den DOS-Prompt etwas besser benutzen zu können, empfehlen wir Ihnen eine *autoxec.bat* wie in *Abb. 12* anzulegen und im Stammverzeichnis der Wartungspartition zu speichern. Beachten Sie jedoch, auch die in der *autoexec.bat* angegebenen Dateien auf ihre Wartungspartition zu kopieren. An der *config.sys* des Wartungssystems sind im allgemeinen keine Änderungen nötig, da *bootos2* die Original-*config.sys* Ihrer Betriebssystempartition als Vorlage verwendet.

Die weitere Ausstattung des Wartungssystems bleibt nun nur noch Ihnen, Ihren Ansprüchen und natürlich Ihrer Phantasie überlassen. ❑

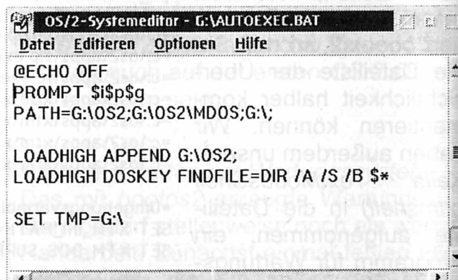


Abb. 12: Beispiel für eine autoxec.bat

Tips & Tricks

In diesem Abschnitt des Know how Teils *der* OS/2 Only! *finden Sie stets Tips & Tricks zu OS/2. Wenn Sie selber einen Trick auf Lager haben, lassen Sie ihn uns einfach zukommen: Per eMail, Post oder auch per Fax. Veröffentlichte Beiträge belohnen wir je nach Umfang mit einem Honorar von DM 25,00 bis DM 50,00.*

Software

Joliet-Unterstützung für OS/2 aktivieren (S002)

Bekannt ist, daß OS/2 die Joliet-Unterstützung für lange Dateinamen auf CD-ROMs bietet, wenn ein entsprechendes Fixpack installiert wurde (für Warp 4 mindestens Fixpack 4). Allerdings ist die Joliet-Unterstützung standardmäßig nicht aktiviert. Um in ihren Genuß zu kommen, übergibt man dem CD-Dateisystem CDFS.IFS zusätzlich den Parameter /W. Achten Sie also darauf, daß die IFS-Anweisung für das CD-Dateisystem so aussieht:

```
IFS=CDFS.IFS /W
```

Alle anderen Parameter, die Sie sonst noch angeben, fügen Sie der Anweisung wie gewohnt hinzu. Nach einem Systemneustart steht Ihnen dann die Joliet-Unterstützung zur Verfügung. Leider gibt das System beim Starten keine Statusmeldung darüber aus, ob die Unterstützung aktiviert ist oder nicht.

Mit dem RSJ CD-Writer Dateisystem ohne Lockteiber arbeiten (S003)

Wir haben bereits dargestellt, wie man mit dem RSJ CD-Writer ohne den mitgelieferten Locktreiber auskommt, und wie man auch mehrere Recorder mit dem Dateisystem betreiben kann. Uns wurde jedoch von Schwierigkeiten mit ähnlichen Vorgehensweisen berichtet, wobei das Dateisystem die folgende Meldung ausgibt: *Die angegebene Einheit konnte nicht gefunden werden.*

Dieses Problem läßt sich nur beseitigen, wenn man dem Treiber *os2aspi.dmd* in der *config.sys* den Parameter */all* übergibt:

```
BASEDEV=OS2ASPI.DMD /ALL
```

Der Treiber *os2aspi.dmd* stellt ein *ASPI (Advanced SCSI Programming Interface)* für SCSI-Gerätetreiber zur Verfügung. Grundsätzlich kann man *os2aspi.dmd* zwei Parameter übergeben, nämlich */share* und */all*. Man legt damit fest, wie Treiber auf unterschiedliche Geräte zugreifen können. Mit dem Parameter */share* werden Befehle an ein SCSI-Gerät nur dann weitergegeben, wenn dieses Gerät z.Zt. nicht von einem bestimmten Treiber (oder auch: Einheintenmanager) benutzt wird. Durch den Parameter */all* werden SCSI-Befehle *immer* und an *jedes* Gerät am Bus weitergeleitet. Greifen verschiedene Einheitenmanager auf ein Gerät zu, kann es daher mit dem Parameter */all* zu Synchronisationsproblemen kommen. Die Verwendbarkeit dieses Parameters ist daher auch von der Systemkonfiguration abhängig. Allerdings konnten wir auf unseren Testsystemen keinerlei Schwierigkeiten mit dem Parameter */all* feststellen, womit die Angabe des Parameters das oben geschilderte Problem beseitigen dürfte, ohne ein neues zu schaffen.

Probleme mit Soundblaster-Karten (S004)

Sortiert man seine *config.sys* und ordnet dabei die Anweisungen, mit denen man

Gerätetreiber in das System einbindet, kann es vorkommen, daß Soundkarten, die den *SoundBlaster 32/64 AWE (1.1.b)-Treiber* benutzten, z.B. die *AWE64 Value*, nicht mehr erkannt werden. Dieses Problem kann man beheben, indem man den Soundkartentreiber *vor* dem Maustreiber lädt.

Jörg Sievers, Hamburg

Cross-Update auf OS/2-Programme (S005)

Häufig liegt Hardware das eine oder andere Programm zur Nutzung des Gerätes bei, z.B. Bildbearbeitung und OCR-Software beim Scanner, Fax-Programm beim Modem, usw., leider jedoch stets nur Versionen für die Betriebssysteme von Microsoft - und Applikationen für Win95/98 oder NT laufen bekanntlich nicht unter OS/2. Kaum ein Programm ist auch als OS/2-Version desselben Herstellers erhältlich, so daß ein 1:1-Umtausch in der Regel nicht machbar ist. So sieht sich mancher gezwungen, zähneknirschend die Vollversion eines OS/2-Programmes zu besorgen. Es gibt dafür aber noch eine andere, von mir schon mehrfach praktizierte Lösung:

Viele Hersteller von Programmen sind gerne bereit, sogenannte Cross-Updates von einem Programm eines anderen Herstellers auf das eigene Produkt zu gewähren (so bekommen sie dann neue Kunden, die sich das nächste, dann reguläre Update sicher nicht entgehen lassen werden - sofern dessen Qualität in Ordnung ist.) Dabei ist zu beachten, daß der Preis für dieses Cross-Update oft dem eines regulären Updates von einer älteren Version entspricht (etwa 1.0 auf 2.1) und damit teurer ist als das letzte verfügbare Update (z.B. von Version 2.0 auf 2.1). Allerdings ist es immer noch deutlich billiger als der Neukauf einer Vollversion - bei meiner letzten Aktion (Impos/2) war es immerhin die Hälfte!

Für diesen Mehrpreis sollte man sich nicht nur die CD, sondern auch gleich das zugehörige Handbuch und die Lizenznummer geben lassen! Nicht jede Online-Hilfe kann ein Handbuch in Papierform ersetzen und die Lizenznummer braucht man ja für das nächste »echte« Update. Also unbedingt den Händler nach einem Cross-Update auf ein OS/2-taugliches Programm fragen. Es gibt für nahezu alle Standard-Aufgaben zumindest gleichwertige Programme als OS/2-Version.

Martin Gratz, Kiel

Hardware

Scannen mit dem UMAX ASTRA 1220S (H002)

Der Betrieb eines UMAX ASTRA 1220S SCSI-Flachbettscanners ist unter OS/2 beim Einsatz von *CFM-Twain 5.10* möglich. Hardwareseitig muß der Scanner an einen OS/2-tauglichen SCSI-Adapter angeschlossen werden (etwa den Adaptec 2940), da der mit dem Scanner ausgelieferte SCSI-Adapter von OS/2 nicht unterstützt wird. Beim Kauf des Scanners sollte man sich außerdem gleich noch ein passendes Adapterkabel (für den Adaptec 2940 25pol. Sub-D auf 50pol. Mini-Sub-D) besorgen.

Nach der Installation von *CFM-Twain* erkennt die Software jedoch den Scanner nicht, obwohl die beim Rechnerstart mit Strom versorgte Hardware vom BIOS des SCSI-Adapters erkannt wurde. Man kann dieses Problem beheben, indem man in der Datei *scanner.ini* von *CFM-Twain*, die nach der Installation im Verzeichnis:

\os2\twain\cfm50\

zu finden ist, mit Hilfe eines Editors einen zusätzlichen Eintrag vornimmt:

```
[ASTRA 1220S]
Driver=UMAX
ParameterFile=A1220S
```

Dabei ist peinlich genau auf Groß- und Kleinschreibung zu achten. Nach einem Neustart des Systems wird der Scanner anstandslos erkannt.

Die Qualität des gescannten Bildes ist überraschend gut: Bei voller Auflösung sieht man sogar die Struktur des Drukkes einer Hochglanz-Illustrierten! (Ein Blick durch eine Lupe bestätigt die Struktur). Es ist jedoch zu empfehlen, den Arbeitsspeicher des Systems entsprechend aufzurüsten: Bei mehr als 200 MByte Größe eines A4-Scans bei voller Auflösung und Echtfarben war die Festplatte einige Minuten ganz schön am Rotieren. Da ist eine Erhöhung des Anfangswertes der *swapper.dat* auf 256 MByte in der *config.sys* schon aus Performance-Gründen angesagt, aber nur eine Kompromiß-Lösung, so daß man für ernsthafte Anwendungen an einer Investition in RAM nicht vorbeikommt.

Martin Gratz, Kiel

Kommandozeile

Kommandozeile anpassen (K002)

Wenn man gerne mit der Kommandozeile arbeitet, sollte man sich den Prompt ein wenig den eigenen Bedürfnissen anpassen.

Zunächst wäre da die Farbe des Hintergrundes. Die Ausgabe mit weißer Schrift auf schwarzem Hintergrund ist nicht optimal. Angenehmer wäre schwarze Schrift auf einem weißen Hintergrund. Hat man außerdem eine hohe Bildschirmauflösung gewählt, so wäre ein Fenster, das etwas größer als die Standardmaximalgröße ist, wünschenswert. Idealerweise sollte es höher sein, um möglichst viele Zeilen Text dartstellen zu können. Um beides zu erreichen, schreibt man sich eine kleine Batchdatei mit folgendem Inhalt:

```
@echo off
mode 90,40
echo ←[0;30;47;5m
cls
```

Mit dem Mode-Befehl legt man die Höhe des Fensters auf 90 Zeilen fest. Sie können hier Einstellungen von 2 bis 102 Zeilen vornehmen. Die Spaltenzahl erhöhen wir mit dem Wert 40 um 15 Spalten (Standard ist ein Fenster mit 80 Zeilen und 25 Spalten). Bei einer Auflösung von 1024 x 768 Bildpunkten verfügt man so über ein angenehm großes Fenster. Um die richtigen Abmessungen der Fenster zu erhalten, müssen Sie noch eine passende Schriftartgröße wählen. Dazu öffnen Sie das Systemmenü des Fensters und wählen den Menüpunkt *Schriftartgröße...* Für unser Beispiel wählen Sie im erscheinenden Dialog *Schriftartgröße angeben* eine Schriftgröße von 18 x 8 in der Listbox *Größe (Bildelemente)*. Durch die Escape-Sequenz:

```
echo ←[0;30;47;5m
```

legen wir fest, daß der Fensterhintergrund weiß und die Schrift darauf schwarz sein soll (den ← erhalten Sie mit Alt+27). Speichern Sie diese Datei in einem Verzeichnis ab, das im PATH steht, um sie stets aufrufen zu können. Möchten Sie, daß die Batchdatei bei jedem Start eines OS/2-Fensters ausgeführt wird, setzen Sie sie als Parameter der Umgebungsvariable OS2_SHELL:

```
SET OS2_SHELL=C:\OS2\CMD.EXE
                    /K c:\os2\mywin.cmd
```

Workshop DLLs, Teil 2

Im letzten Teil unseres Workshops haben wir gelernt, wie man DLLs schreibt und sie verwendet. In diesem Teil werden wir uns mit einem weiteren interessanten Einsatzzweck von DLLs beschäftigen, nämlich der REXX-Makrounterstützung bzw. der Erweiterung von REXX durch neue DLLs.

Um dies zu zeigen, wollen wir einen unserer selbst definierten Buttons unseres Beispielprogramms mit einer Makrofunktion belegen: Drückt man auf den Button, soll das Programm ein REXX-Makro ausführen. Damit wir mit dem Makro auch komfortabel arbeiten können, schreiben wir später noch eine kleine Programmbibliothek, um die Funktionalität von REXX zu erweitern.

Wozu Makros und warum REXX?
So mancher mag sich fragen, warum man seine Programme durch eine Makrounterstützung erweitern sollte. Wir werden alle Aspekte der Entwicklung eines Makrosupports in diesem Artikel nicht betrachten können. Vielmehr untersuchen wir die Abschnitte, die für den Bereich der DLL-Programmierung interessant sind. Dennoch sind Sie mit den hier wiedergegebenen Informationen in der Lage, Ihre Programme durch die Möglichkeit der Makrodefinition noch benutzerfreundlicher zu gestalten.

Das Konzept des Beispielprogramms
Der Beispielcode zu diesem Workshop wurde noch einmal grundlegend überarbeitet. Sie finden ihn mit Erscheinen dieses Bandes nun auch endlich auf unserer Homepage zum Download. Das kleine Programm *XTest* demonstriert Ihnen mit einem einfachen Beispiel, wie DLLs benutzt werden, und wie das Zusammenspiel zwischen C-Applikation und REXX funktioniert. *XTest* ist eine abgespeckte Variante des gleichnahmigen Programms, das wir zum Test neuen Programmcodes für *Xelia* verwenden. Mit Beginn der C- und Pascal-Programmierworkshops wird Ihnen dieses Programm in seinem vollen Umfang auf der Dezember-CD zur Verfügung stehen. Sie erhalten dann mit *XTest* Beispielcode zu den unterschiedlichsten Bereichen der OS/2-Programmierung. Die Version von *XTest*, die Ihnen jetzt vorliegt, macht von unseren DLLs Gebrauch, indem in einem Fenster drei der neu definierten Buttons und eine Statuszeile erstellt werden. Zusätzlich lädt *XTest* in ein MLE eine Datei im ASCII-Format, die durch ein in REXX verfaßtes Makro, das über den Button *Ausführen* gestartet wird, in HTML konvertiert wird. Anschließend stellt *XTest* den veränderten Dateiinhalt im MLE dar. Durch Austausch des Makros gegen ein anderes wird der modulare Charakter und die leichte Erweiterbarkeit des Programms durch REXX klar. Um die ASCII-Datei mit einer C-Funktion konvertieren zu können, schreiben wir einen HTML-Filter in C, den wir von unserem REXX-Makro aus aufrufen. So bleibt der REXX-Code einfach und läßt sich flexibel erweitern. Die Programmoberfläche zeigt Abbildung 2. In Abbildung 1 finden Sie die Arbeitsweise, die dem Programm zugrunde liegt.

Wie man REXX verwenden kann
Es gibt grundsätzlich zwei verschiedene Arten, REXX und C miteinander zu kombinieren: Man kann eine REXX-Prozedur von einem C-Programm aus aufrufen; oder man schreibt eine Funktionsbibliothek in C, die dazu dient, die exportierten Funktionen innerhalb einer REXX-Prozedur aufzurufen. Im ersten Fall erweitert man die in C geschriebene Anwendung mit REXX; im zweiten Fall die REXX-Prodzedur durch C-Funktio-

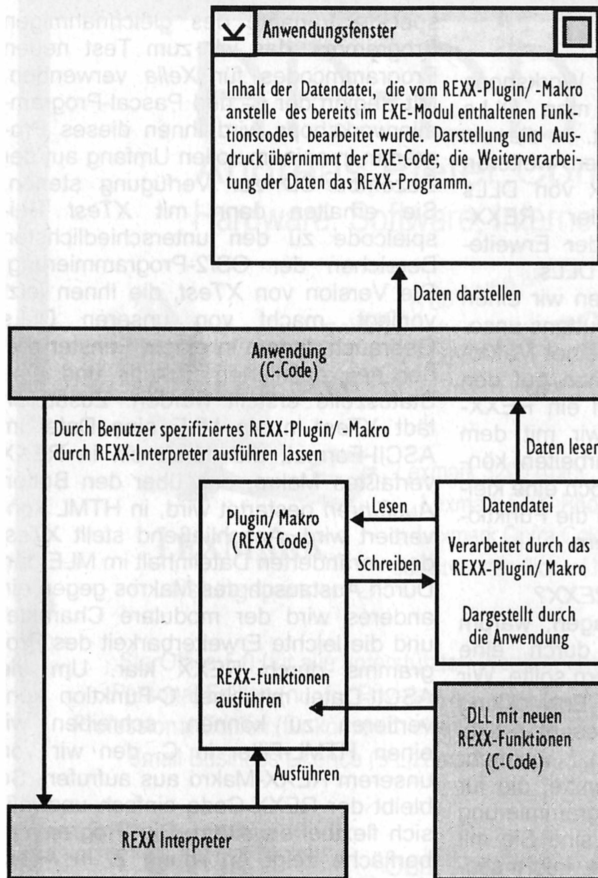


Abb. 1: Struktur einer Anwendung, die von REXX Gebrauch macht.

nen. Wir werden in diesem Artikel beide Möglichkeiten betrachten. Zur Verdeutlichung der Struktur einer Anwendung, die von REXX Gebrauch macht, dient Abb. 1.

Im allgemeinen wird die Anwendung ein PM-Programm sein, das die zu verarbeitenden Daten optisch ansprechend in Fenstern darstellt und für die einzelnen Bearbeitungsprozesse geeignete Dialoge zur Verfügung stellt. Ein- und Ausgabefunktionen werden also über die PM-Schnittstelle vorgenommen. Die eigentliche Verarbeitung der Daten (oder deren Weiterverarbeitung) über das REXX-Programm.

Ein Makro dient dazu, immer wiederkehrende Arbeitsabläufe in einem Skript zusammenzufassen, um sie später automatisch ablaufen zu lassen. Ein Plugin ist bereits ein ganzer Programmteil, der die Basisanwendung um neue Funktionen erweitert. Ein Import-Export-Filter wäre z.B. eher ein Plugin als ein Makro. Eine REXX-Prozedur, die nacheinander Windowsbitmaps in GIF-Bitmaps umwandelt, wäre ein Makro.

Hinsichtlich der zu benutzenden Schnittstellen zwischen dem C- und REXX-Teil der Anwendung gibt es jedoch keine Unterschiede, ob man eine REXX-Prozedur schreibt, um ein Programm mit neuen Funktionen zu erweitern, wie wir es mit dem Skript in Abb. 2 tun, oder ob man unterschiedliche Bearbeitungsmethoden automatisiert mit Hilfe eines geeigneten REXX-Skriptes ausführt. Um beide Programmteile miteinander kommunizieren zu lassen, benutzen wir in unserem Beispiel eine Datei. Dateien lassen sich bedeutend einfacher verwalten als komplexe Strukturen im Speicher. Zudem ist das Dateimanagement sowohl C- als auch REXX-seitig schnell zu beherrschen. Mit der PM-Anwendung kann man z.B.

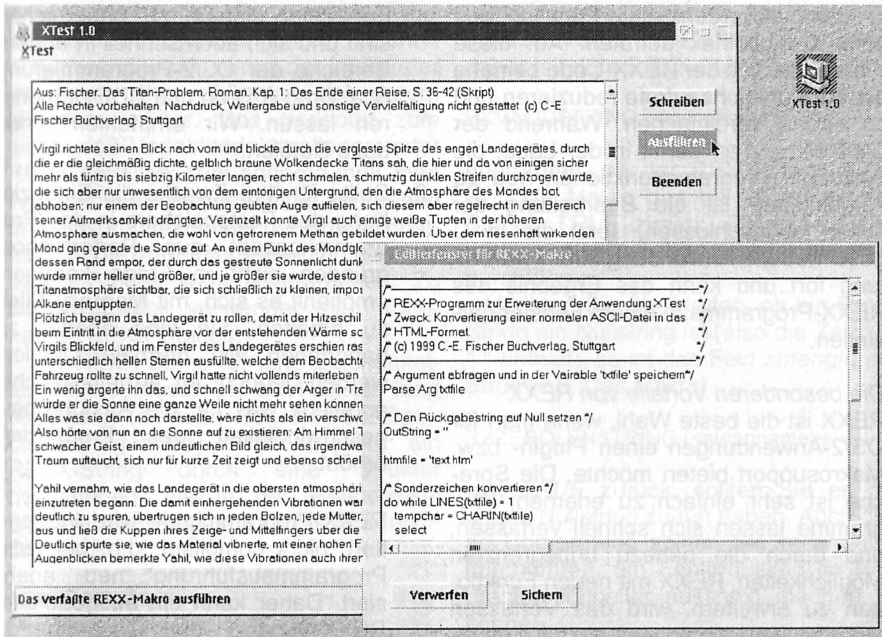


Abb. 2: Die Oberfläche des Beispielprogramms mit geöffnetem Makro-Editierfenster.

einen Text erzeugen, den man von der REXX-Erweiterung des Programms in ein anderes Dateiformat konvertieren läßt. Die konvertierte Version soll dann durch die Benutzerschnittstelle angezeigt werden. Unser Beispielprogramm arbeitet nach diesem Prinzip. Führen Sie es einmal aus, um sich die Arbeitsweise zu verdeutlichen. Mit dem Button *Schreiben* kann der Benutzer den REXX-Code verändern oder durch eine völlig andere Version ersetzen. Auf diese Weise wird die Funktionalität des Programms ebenfalls verändert, ohne daß man es neu hätte kompilieren müssen, da REXX-Code interpretiert wird. Man sollte deshalb jedoch nicht annehmen, die Programmausführung würde dadurch sehr verlangsamt. Man kann auch REXX-Algorithmen optimieren. Die Codequalität entscheidet in hohem Maße über die Ausführungsgeschwindigkeit eines Programms, gleich in welcher Sprache es geschrieben wurde. Sollten geringe Geschwindigkeitseinbußen spürbar sein, so kann man bestimmte Funktionen auch in C programmieren und sie REXX über eine DLL zur Verfügung stellen.

Verfolgen wir den Ablauf des Programms in Abb. 1. Der Benutzer wählt zunächst die Funktion aus, welche den REXX-Code ausführen soll. Dieser befindet sich in der Regel in einer gesonderten Datei auf der Festplatte. Um das Skript zu starten, ruft die Anwendung den REXX-Interpreter auf, welcher den REXX-Code mit den von der Anwendung übergebenen Startparametern ausführt. Die gemeinsame Schnittstelle zwischen REXX- und C-Anwendung ist dabei eine Datendatei. Der REXX-Code nimmt die zuvor definierte Bearbeitung an dieser Datei vor.

Er kann außerdem eine Funktion aus einer C-Bibliothek aufrufen. Auf diese Weise läßt sich der REXX-Code beinahe bis auf Funktionsaufrufe reduzieren und so enorm vereinfachen. Während der Laufzeit ist man damit in der Lage, die einzelnen Programmfunktionen neu zu kombinieren. Ist die Bearbeitung der Daten abgeschlossen, setzt die in C geschriebene Anwendung ihre Verarbeitung fort und kann das Ergebnis des REXX-Programms in einem Fenster darstellen.

Die besonderen Vorteile von REXX

REXX ist die beste Wahl, wenn man für OS/2-Anwendungen einen Plugin- bzw. Makrosupport bieten möchte. Die Sprache ist sehr einfach zu erlernen, Programme lassen sich schnell verfassen, und durch die nahezu unbegrenzten Möglichkeiten, REXX mit neuen Funktionen zu erweitern, wird das Verfassen eigenen Programmcodes auch für ungeübte Benutzer möglich. Man kann dabei so weit gehen, wichtige Programmteile nur mit REXX zu entwickeln. Wenn Sie eine Dokumentation der verwendeten Übergabeparameter an den REXX-Interpreter durch das Programm und die Ihrem REXX-Support zugrundeliegenden Schnittstellen (in unserem Falle eine Datei) dokumentieren, lassen sich Ihre Anwendungen nahezu jedem Einsatzzweck anpassen - vorausgesetzt, Sie bieten auch entsprechende Bibliotheken zur Erweiterung von REXX. Wir haben diese Möglichkeit für unsere Software sehr weitreichend ausgeschöpft, um eine einfache Softwareentwicklung zu ermöglichen. Beachten Sie für die Implementierung des REXX-Supportes grundsätzlich:

1. Benutzen Sie Schnittstellen zwischen C-Anwendung und dem REXX-Programm, die von jedem Entwickler einfach zu beherrschen sind und sich auch schnell in andere Bereiche der OS/2-Programmierung (v.a. Drag & Drop und DDE) integrieren lassen. Wir empfehlen Ihnen dazu Dateien.

2. Sollte für Ihre Zwecke die Einbeziehung des Festspeichers in das Programmkonzept aus Performancegründen nicht in Frage kommen, empfiehlt es sich, mit **Memory Files** zu arbeiten. Sie sind ebenso zu handhaben wie normale Dateien, werden jedoch im Arbeitsspeicher angelegt, was i.a. die Ausführungsgeschwindigkeit des Programms erhöht.

3. REXX-Programmcode wird interpretiert, nicht compiliert, d.h. bei jeder Programmausführung neu analysiert. Daher kann ein ausgedehnter REXX-Support die Gesamtperformance Ihrer Anwendung verringern. Verringern Sie in diesem Fall den Anteil des REXX-Codes in Ihrer Anwendung, und kapseln Sie rechenintensive Funktionen in DLLs aus. REXX-Code, der fast nur aus Funktionsaufrufen besteht, bedingt keine spürbaren Geschwindigkeitsverluste. Der modulare Charakter Ihrer Anwendung bleibt jedoch erhalten.

Wir werden im folgenden darstellen, wie REXX in eine mit C entwickelte PM-Anwendung integriert wird. Dabei wird zunächst untersucht, wie man ein bestehendes Programm mit REXX, daran anschließend wie man REXX mit einer neuen DLL erweitert.

Über Argumente und Rückgabewerte

REXX arbeitet nur mit Strings (Zeichenketten), auch wenn es sich um numeri-

sche Werte handelt, die verarbeitet werden sollen. Die jeweils nötigen Datentypenumwandlungen nimmt der Interpreter intern vor, ohne daß man sich darum kümmern müßte. Das bedeutet zum einen, daß Variablendeklarationen unter REXX entfallen; zum anderen muß man Argumente als Zeichenketten übergeben und entsprechende Mechanismen schaffen, um die vom Interpreter zurückgegebenen Strings weiterzuverarbeiten. Zu diesem Zweck verwendet man sogenannte REXX-Strings.

Ein REXX-String ist eine Zeichenkette, die theoretisch bis zu 4 GByte lang sein kann (s. auch Bd. 1: OS/2-Speichermanagement). Unter C wird ein REXX-String durch eine Struktur beschrieben, die in der Headerdatei *rexxsaa.h* definiert ist. Die Strukturdefinition lautet:

```
typedef struct
{
  ULONG strlength; /* Stringlänge */
  PCH      strptr;      /* Zeiger auf den String */
} RXSTRING;

/* Zeiger auf einen REXX-String */
typedef RXSTRING *PRXSTRING;
```

Um mit REXX-Strings zu arbeiten, sind im oben genannten Header noch ein ganze Reihe Makros definiert und natürlich die Funktionsprototypen des REXX-APIs. Wir können hier nicht auf alle REXX-String-Makros eingehen, auch nicht auf alle APIs, das bleibt unseren REXX- und C-Workshops vorbehalten. Hier sollen nur für unseren Zweck wichtige Makros bzw. Funktionen betrachtet werden.

REXX-Strings können einen bestimmten Wert haben, wobei dieser Wert auch ein Nullstring sein kann. Ebenso ist es möglich, daß ein REXX-String auch ganz leer ist. In diesem Fall ist die Zeigervariable *strptr* der Struktur NULL. Die Arbeit mit einem solchen REXX-String würde zu einem Abbruch des Programm mit einem SYS 3175 führen. Sie können daher mit dem REXX-String-Makro:

```
RXVALIDSTRING (<stringname>)
```

überprüfen, ob der REXX-String gültig ist. Das Makro gibt bei Gültigkeit des Strings *TRUE* zurück. Derartige Überprüfungen sollten Sie zur Vorsicht immer durchführen.

Möchten Sie überprüfen, ob ein Rexx-String ein Nullstring ist (also die Zeichen " " enthält), so ist das Feld *strlength* der Struktur 0. Das Makro:

```
RXZEROLENSTRING (<stringname>)
```

gibt *TRUE* zurück, sollten Sie es mit einem Nullstring zu tun haben. Auch diese Überprüfung sollten Sie regelmäßig durchführen, um Fehlerquellen im Programm schneller ausfindig machen zu können.

Grundsätzlich sollten Sie die C-Bibliotheksfunktionen zur Manipulation von Strings nicht verwenden, um REXX-Strings zu bearbeiten. Benutzen Sie dazu besser die Funktionen aus der Bibliothek *rxstring.lib*, die Sie zusammen mit der Bibliothek *rexx.lib* mit Ihrem Programmcode linken müssen.

! *Die beiden Libraries werden nicht standardmäßig mit Ihrem Programmcode gelinkt. Sie müssen daher beide Bibliotheken dem Linker explizit mitteilen.*

Da wir als gemeinsame Schnittstelle zwischen dem REXX- und C-Teil unserer Anwendung Dateien verwenden, können wir uns in unserem Beispiel darauf beschränken, lediglich den Dateinamen, also normale Zeichenketten, in einem vorbereiteten REXX-String zu übergeben und einen Dateinamen als Rückgabewert unseres REXX-Pro-

gramms zu verarbeiten. Für viele Zwekke reicht eine derartige Vorgehensweise aus. Ab Band 4 werden wir uns intensiv mit REXX-Strings auseinandersetzen.

Eine Anwendung mit REXX erweitern
Wir betrachten zunächst den Fall, eine REXX-Prozedur von einem C-Programm aus aufzurufen. In unserem Beispiel soll eine ASCII-Datei in eine HTML-Datei konvertiert werden. Die Konvertierung soll ein REXX-Programm übernehmen. Um den Code einfach zu halten, beschränken wir uns lediglich auf die Umsetzung der Umlaute und des ß in das HTML-Format. Den REXX-Code können wir mit der Editierfunktion unseres Beispielprogramms oder auch einfach mit dem Systemeditor verfassen. *Codebeispiel 1* zeigt die REXX-Prozedur. Anschließend rufen wir den REXX-Code von der C-Applikation mit einer

Codebeispiel 1: Die REXX-Prozedur zur Dateiformatkonvertierung

```
/*--------------------------------------------------------------------------------*/
/* REXX-Programm zur Erweiterung der Anwendung XTest  */
/* Zweck: Konvertierung einer normalen ASCII-Datei in das  */
/* HTML-Format.                                            */
/*--------------------------------------------------------------------------------*/
/* Argument abfragen und in der Vairable 'txtfile' speichern*/
PARSE Arg txtfile

/* Den Rückgabestring auf Null setzen */
OutString = ''

htmfile = 'text.htm'

/* Sonderzeichen konvertieren */
do while LINES(txtfile) = 1
   tempchar = CHARIN(txtfile)
   select
   when tempchar = "ä" then call CHAROUT htmfile, "&auml;"
   when tempchar = "ö" then call CHAROUT htmfile, "&ouml;"
   when tempchar = "ü" then call CHAROUT htmfile, "&uuml;"
   when tempchar = "Ä" then call CHAROUT htmfile, "&Auml;"
   when tempchar = "Ö" then call CHAROUT htmfile, "&Ouml;"
   when tempchar = "U" then call CHAROUT htmfile, "&Uuml;"
   when tempchar = "ß" then call CHAROUT htmfile, "&szlig;"
   otherwise
     call CHAROUT htmfile, tempchar
   end
end

call LINEOUT (htmfile)
call LINEOUT (txtfile)

/* Den Rückgabestring auf Dateiname setzen */
OutString = htmfile

/* Den Rückgabestring zurückgeben und das REXX-Programm beenden*/
Return OutString
```

entsprechenden Funktion aus dem REXX-API auf (*s. Codebeispiel* 2).

Betrachten wir zunächst das REXX-Programm. Wie man sieht, erwartet es als Argument den Namen einer Datei, die den zu konvertierenden Text enthält. Mit:

```
PARSE Arg txtfile
```

wird der Parameter, welcher der REXX-Prozedur übergeben wurde, in der Variable *txtfile* gespeichert. Anschließend wird mit:

```
OutString = ''
```

der Rückgabestring vorbereitet und auf NULL gesetzt und mit:

```
htmfile='text.htm'
```

eine Dateivariable namens htmfile initialisiert. In diese Datei schreibt die REXX-Prozedur in der folgenden Schleife alle Zeichen der Originaldatei, deren Name sich in der Variablen txtfile befindet. Innerhalb der while-Schleife wird die Originaldatei mit Hilfe der Anweisung:

tempchar = CHARIN(txtfile)

zeichenweise ausgelesen, wobei die Umlaute und das ß durch die entsprechenden HTML-Codes ersetzt werden. Alle anderen Zeichen werden durch das die Schleife abschließende *otherwise*:

```
otherwise
  call CHAROUT htmfile, tempchar
end
```

ohne Veränderung in die neue HTML-Datei übernommen. Zum Schluß werden beide Dateien geschlossen und der Name der erzeugten HTML-Datei in die Variable *OutString* geschrieben. Mit:

```
Return OutString
```

wird das REXX-Skript beendet und der Rückgabewert ausgegeben, den wir durch das C-Programm weiterbearbeiten werden.

Das PM-Programm übernimmt die Verwaltung der Programmoberfläche und startet das REXX-Skript. Dazu gehören neben dem Erzeugen der REXX-Übergabeparameter und dem eigentlichen Aufruf des REXX-Interpreters auch die korrekte Bearbeitung des Rückgabewertes der REXX-Prozedur. Da das REXX-Skript durch das Drücken eines Buttons ausgeführt werden soll, befindet sich der für die REXX-Unterstützung relevante Code innerhalb des Abschnittes mit dem Programmcode zur Bearbeitung der WM_COMMAND-Nachricht.

Für diesen Bearbeitungsabschnitt definierte Variablen sind zwei REXX-Strings *rxstrArg* und *rxstrRet*. Der erste String nimmt die Parameter auf, die der REXX-Prozedur übergeben werden. In unserem Falle ist dies nur ein Parameter, der den Namen der Textdatei enthält, die konvertiert werden soll. Der zweite String enthält den Rückgabewert der REXX-Prozedur, den wir mit der Anweisung RETURN ausgeben. Fehler des Interpreters finden wir in der Variablen *rc*. Mit *chFileName* definieren wir eine Variable, die den Namen der zu konvertierenden Datei aufnimmt, welche *text.xdf* heißt und im gleichen Verzeichnis wie die ausführbare Programmdatei liegt.

Der folgende Code zeigt den Aufruf der REXX-Routine. Zunächst wird das Feld *strlength* des REXX-Strings, der den Rückgabewert enthalten soll, mit der Anweisung:

```
rxstrRet.strlength = 0L;
```

auf 0 gesetzt. Damit wird der REXX-Interpreter dazu veranlaßt, einen REXX

Codebeispiel 2: Den REXX-Interpreter aufrufen

```
MRESULT EXPENTRY XTestMainClientWndProc (HWND hwnd ... )
{
// andere Nachrichten bearbeiten...

case WM_COMMAND:
  switch (SHORT1FROMMP (mpParam1))
  {
    case ID_XTEST_MAINWND_BTN2:
    {
      RXSTRING rxstrArg;
      RXSTRING rxstrRet;
      APIRET rc;
      SHORT rxrc;
       CHAR *chFileName = "text.xdf";

       XeliaSetStatusbarText ("Führe REXX-Makro aus...");
       rxstrRet.strlength = 0L;

       MAKERXSTRING (rxstrArg, chFileName, strlen (chFileName));

      if(RXVALIDSTRING (rxstrArg))
      {
        // An dieser Stelle wird das REXX-Makro aufgerufen
       rc=RexxStart((1,
                    &rxstrArg,
                    "HTMASCI.XPI",
                    0,
                    "XPI",
                    RXSUBROUTINE,
                    0,
                    &rxrc,
                    &rxstrRet );

       XeliaMLEClearTextPanel (hwndMLE);
       XeliaMLELoadFile (hwndMLE, (PSZ) rxstrRet.strptr);


       DosFreeMem(rxstrRet.strptr);
       XeliaSetStatusbarText ("Bereit");
     } // endif
    }
    break;

  //weitere WM_COMMAND-Nachrichten bearbeiten...
} // endswitch

//weitere Nachrichten bearbeiten...

} // Ende XTestMainClientWndProc
```

String mit *DosAllocMem* zu allokieren, der groß genug ist, um den Rückgabewert aufzunehmen. Wir geben diesen String nach der Verarbeitung des Rückgabewertes mit *DosFreeMem* wieder frei.

! *Beachten Sie immer, den vom Interpreter allokierten Speicher wieder mit* DosFreeMem *freizugeben.*

Im nächsten Schritt wird der REXX-String, der den Übergabeparameter an die REXX-Routine aufnehmen soll, allokiert und auf seine Gültigkeit hin überprüft. Um REXX-Strings zu allokieren und zu initialisieren, benutzen Sie das Makro:

```
MAKERXSTRING (rxstrArg, chFileName,
                    strlen (chFileName));
```

wobei der erste Parameter der Name des REXX-Strings ist, den wir bereits definiert haben. Der zweite Parameter bezeichnet die Zeichenkette, die der REXX-String aufnehmen soll und der dritte Parameter die Länge dieses Strings. Verwenden Sie stets nur dieses Makro, um REXX-Strings zu allokieren. Anschließend überprüfen Sie mit dem Makro:

```
RXVALIDSTRING (rxstrArg)
```

ob das Allokieren und Initialisieren des Strings erfolgreich war. Das Makro gibt TRUE zurück, sollte dies der Fall sein. Wir überprüfen mit dem Makro den REXX-String un können nun die REXX-Routine aufrufen. Dazu dient die Funktion *RexxStart*, welche Sie immer nach dem im folgenden erklärten Muster aufrufen, um REXX-Skripte von einer Applikation aus zu starten. Bedenken Sie noch einmal, daß wir nur die Parameter benutzen, die für unseren Zweck von Bedeutung sind. Die ausführliche Erklärung dieser Funktion bleibt dem C-Programmierkurs überlassen. Der hier beschriebene Aufruf der Funktion ist jedoch der gebräuchlichste, wenn man einzelne REXX-Skripte von einer Anwendung aus starten möchte. Der Aufruf sieht wie folgt aus:

```
rc=RexxStart(1,
            &rxstrArg,
            "HTMASCI.XPI",
            NULL,
            "XPI",
            RXSUBROUTINE,
            0,
            &rxrc,
            &rxstrRet );
```

Der erste Parameter von *RexxStart* steht für die Anzahl der Parameter, die dem REXX-Interpreter übergeben werden. Unser Beispielprogramm arbeitet nur mit einem Parameter, daher steht hier eine 1. Möchten oder müssen Sie mehr als nur einen Parameter verwenden, so definieren Sie ein Array von REXX-Strings:

```
RXSTRING rxstrArg[3];
```

Anschließend initialisieren Sie die einzelnen Parameter. Wir nehmen an, Sie haben numerische Werte zuvor in Zeichenketten umgewandelt und in entsprechende Variablen kopiert, etwa in *chParam1*, *chParam2* und *chParam3*. Das Erzeugen des REXX-String-Arrays sieht dann so aus:

```
MAKERXSTRING (rxstrArg[0], chParam1,
                    strlen (chParam));
MAKERXSTRING (rxstrArg[1], chParam2,
                    strlen (chParam));
MAKERXSTRING (rxstrArg[2], chParam3,
                    strlen (chParam));
```

Beachten Sie in diesem Fall jedoch, daß Sie der Funktion *RexxStart* die entspre-

chende Anzahl der für die REXX-Prozedur nötigen Parameter übergeben. Im REXX-Skript verarbeiten Sie mehr als nur einen Parameter ebenfalls mit der Anweisung *Parse Arg* und fügen einfach Variablen in ausreichender Anzahl an:

```
PARSE Arg param1, param2, param3
```

Der zweite Parameter von *RexxStart* bezeichnet das REXX-String-Array, das die zur Ausführung des REXX-Skriptes nötigen Argumente enthält. Der nächste Parameter beschreibt den Namen der Datei, in der Sie den REXX-Code abgelegt haben. Den vierten Parameter benötigen wir in diesem Falle nicht. Hier wird ein Array mit zwei REXX-Strings spezifiziert, die für das Auffinden von REXX-Prozeduren dienen, die sich im Arbeitsspeicher befinden. Wir geben hier NULL an. Anschließend wird der Initialisierungswert für die REXX-Anweisung *ADRESS* angegeben. Geben Sie in diesem Falle die Erweiterung des Dateinamens der REXX-Routine an, die Sie geschrieben haben. Der sechste Parameter von *RexxStart* legt fest, wie die REXX-Routine ausgeführt werden soll. Für unsere Zwecke interessant sind RXSUBROUTINE und RXFUNCTION. Im ersten Fall muß die REXX-Routine keinen Rückgabewert ausgeben, im zweiten ist dies notwendig. RXSUBROUTINE bietet damit eine höhere Flexibilität, weshalb wir Ihnen empfehlen, diese Konstante im Funktionsaufruf zu benutzen. Der nächste Parameter wird benötigt, wenn man eigene Exit-Handler benutzt. Dies ist in unserem Beispiel nicht nötig, so daß wir hier NULL angeben. Die letzen beiden Parameter dienen zur Angabe der Variablen, welche die Rückgabewerte der REXX-Routine aufnehmen sollen. Geben Sie hier *die Pointer* auf die entsprechenden Variablen an, die Sie im Deklarationsteil des Programmabschnittes definiert haben.

Durch den so gestalteten Funktionsaufruf wird der REXX-Interpreter aufgerufen und die zuvor abgefaßte REXX-Routine ausgeführt. Nach deren Beendigung können wir den Rückgabewert der Routine bearbeiten. Dieser besteht aus dem Namen der durch das REXX-Programm erzeugten HTML-Datei. Gemäß der zuvor dargestellten Richtlinien soll die PM-Anwendung die Darstellung dieser Datei übernehmen. Also führen wir einen Refresh unseres MLEs mit:

```
XeliaMLEClearTextPanel (hwndMLE);
```

durch und laden anschließend die im zurückgegebenen REXX-String angegebene Datei:

```
XeliaMLELoadFile (hwndMLE,
                 (PSZ) rxstrRet.strptr);
```

Letztlich müssen wir nur noch, wie bereits ausgeführt, den vom REXX-Interpreter allokierten Speicher freigeben:

```
DosFreeMem(rxstrRet.strptr);
```

Damit ist der nötige Programmcode vollständig. Nach diesem Muster können Sie jederzeit und an jeder Stelle Ihres PM-Programms REXX-Unterprogramme ausführen. Das Beispielprogramm demonstriert außerdem eindrucksvoll, wie Sie das Ergebnis der Datenverarbeitung während der Laufzeit des Programms durch Manipulation des REXX-Codes verändern können. Mit den hier gegebenen Anweisungen und mit ein wenig Phantasie sollte es leicht sein, diese Art des REXX-Supportes zu erweitern und zu verbessern.

REXX mit einer eigenen DLL erweitern

Sie bemerken wahrscheinlich, daß die Umsetzung des ASCII-Textes in das HTML-Format recht langsam vor sich

Codebeispiel 3: Aufruf einer neuen REXX-Funktion

```
/*-------------------------------------------------------------------------------*/
/* REXX-Programm zur Erweiterung der Anwendung XTest     */
/* Zweck: Konvertierung einer normalen ASCII-Datei in das     */
/* HTML-Format.                                               */
/*-------------------------------------------------------------------------------*/
/* Argument abfragen und in der Vairable 'txtfile' speichern*/
Parse Arg txtfile

/* Von der DLL xrexx.dll exportierte Funktionen registrieren*/
call RxFuncAdd 'XeliaREXXRegisterFunctions', 'XREXX', 'XeliaREXXRegisterFunctions'
call XeliaREXXRegisterFunctions

/* Den Rückgabestring auf Null setzen */
OutString = ''

/* Die Konvertierung nach HTML vornehmen */
call XeliaREXXASCIToHTML txtfile, 'text.htm'

/* Den Rückgabestring auf Dateiname setzen */
OutString = 'text.htm'

/* Die zuvor registrierten Funktionen wieder deregistrieren */
call XeliaREXXDeRegisterFunctions

/* Den Rückgabestring zurückgeben und das REXX-Programm beenden*/
Return OutString
```

geht, wenn Sie auf den Button *Ausführen* des PM-Programms drücken, was darauf hinweist, daß der Konvertierungsfilter, der vollständig mit REXX geschrieben wurde, interpretiert wird. Wir können die Leistung unseres Programms enorm verbessern, wenn wir den zur Konvertierung entscheidenden Code in C schreiben und ihn als Funktion vom REXX-Skript aus aufrufen. Dabei bleibt der große Vorteil -nämlich das Programm auch während der Laufzeit zu verändern- erhalten; jedoch wird der Anteil des REXX-Codes bis auf den Funktionsaufruf minimiert, was einen erheblichen Geschwindigkeitsvorteil mitsichbringt. Um das zu erreichen, müssen wir eine geeignete DLL schreiben, welche REXX um eine Funktion zum Konvertieren von ASCII-Text in HTML-Text erweitert. Sie können diese Konvertierungsmethode dann auch in ganz normalen REXX-Programmen verwenden. Die folgenden Abschnitte zeigen, wie Sie eine solche DLL schreiben.

Neue Funktionen für REXX

Wenn Sie selbstgeschriebene Funktionen für REXX benutzen wollen, müssen folgende Voraussetzungen erfüllt sein:

1. Der Funktionscode muß sich in einer DLL befinden.
2. Alle Funktionen, die von REXX aus aufrufbar sein sollen, müssen als REXX-Funktionen exportiert werden.
3. Alle Funktionen müssen im REXX-Skript zunächst mit *RexxFuncAdd*

registriert werden, damit sie der Interpreter findet.

Zunächst überlegen wir uns, wie die neue DLL und die REXX-Funktion zur Konvertierung heißen sollen. Wir nennen die Beispiel-DLL *xrexx.dll* und die zu exportierende Konvertierungsmethode *XeliaREXXASCIToHTML*. Diese Funktion benötigt zwei Parameter: Den Namen der Originaldatei und den der HTML-Datei, die erzeugt werden soll. Betrachten wir nun zunächst wieder, wie das neue REXX-Skript aussieht, das vom PM-Programm gestartet wird. *Codebeispiel 3* auf S. 141 zeigt das neue Programm.

Zunächst fragen wir wie im ersten Programm den Übergabeparameter, an die REXX-Routine mit:

```
PARSE Arg txtfile
```

ab. Die nächsten beiden Codezeilen sorgen für die Registrierung der neuen REXX-Funktion:

```
call RxFuncAdd
        'XeliaREXXRegisterFunctions',
        'XREXX',
        'XeliaREXXRegisterFunctions'

call XeliaREXXRegisterFunctions
```

RxFuncAdd ist eine Built-In-Funktion des REXX-Interpreters und wird immer dann aufgerufen, wenn extern -also in DLLs- definierte Routinen unter REXX verwendet werden sollen. Wir registrieren hier die Funktion *XeliaREXXRegisterFunctions*, welche sich neben der eigentlich für uns interessanten Funktion in der DLL *xrexx.dll* befindet, die man *RxFundAdd* als zweiten Parameter ohne die Dateinamenerweiterung angibt. Diese Funktion registriert alle in der DLL enthaltenen, von uns definierten REXX-Funktionen, ohne daß wir *RxFuncAdd* für jede einzelne Methode extra aufrufen müßten. Zwar haben wir es in unserem Beispiel nur mit einer Funktion, die exportiert werden soll, zu tun; aber wir demonstrieren diese Vorgehensweise, damit der DLL-Code auch für umfangreichere Bibliotheken leicht erweiterbar bleibt. Wir beschäftigen uns mit dem genauen Aufbau der Funktion *XeliaREXXRegisterFunctions* im nächsten Abschnitt.

XeliaREXXRegisterFunctions wird in der nächsten Zeile mit

```
call XeliaREXXRegisterFunctions
```

aufgerufen. Danach steht uns die REXX-Funktion zur Konvertierung der Textdatei nach HTML zur Verfügung. Mit:

```
call XeliaREXXASCIToHTML txtfile, 'text.htm'
```

wird die Funktion aufgerufen. Die Verarbeitung der Datei wird dann durch den in unserer DLL *xrexx.dll* definierten Code durchgeführt.

Zum Schluß deregistrieren wir alle Funktionen unserer REXX-DLL mit dem Pendant zu *XeliaREXXRegisterFunctions*:

```
call XeliaREXXDeRegisterFunctions
```

weisen den Namen der erzeugten Datei der Rückgabevariablen *OutString* zu und übergeben diese wie gehabt mit:

```
Return OutString
```

wobei die Routine gleichzeitig beendet wird.

Übrigens müssen am ausführbaren Programmodul *xtest.exe* keinerlei Änderungen vorgenommen werden, um das neue REXX-Skript auszuführen, das wir gerade näher betrachtet haben. Lediglich das Skript selbst und die DLL

xrexx.dll müssen zum Zeitpunkt des Mausklicks auf *Ausführen* vorhanden sein.

Wie man eine REXX-DLL schreibt
Betrachten wir nun die Entwicklung der DLL, die etwas komplizierter ist. Wie Sie sich vorstellen können, ist der C-Code zur Dateiformatkonvertierung nicht sonderlich anders als der zuvor in REXX dargestellte Code. Es gibt jedoch einige Regeln, die beim Schreiben einer REXX-DLL zu beachten sind. Wir stellen Ihnen im folgenden den vollständigen C-Code, den Header und die DEF-Datei der DLL vor. Wenn Sie die sich daran anschließenden Bemerkungen beachten, können Sie den hier vorgestellten Code als Gerüst für Ihre eigenen DLLs verwenden und sollten mit der Entwicklung neuer REXX-Bibliotheken keine Schwierigkeiten haben. Werfen wir zunächst einen Blick auf den Header der DLL. *Codebeispiel 4* zeigt die Datei *xrexx.h*.

Die include-Anweisungen sind Ihnen sicherlich bis auf *INCL_REXXSAA* bekannt. Mit dieser Anweisung binden wir den gesamten REXX-Header des C-Toolkits in die DLL ein. Wir benötigen zwar nicht alle Deklarationen dieser Datei, aber können uns uns die Arbeit für die Beispiel-DLL etwas vereinfachen, wenn wir den ganzen Header einbinden.

Codebeispiel 4: Header-Datei der neuen REXX-DLL "xrexx.h"

```
#define INCL_WIN
#define INCL_DOS
#define INCL_ERRORS
#define INCL_REXXSAA
#include <os2.h>
#include <rexxsaa.h>
#include <memory.h>
#include <malloc.h>
#include <fcntl.h>
#include <ctype.h>
#include <string.h>
#include <stdlib.h>
#include <stdio.h>

// Alle Funktionen, die exportiert werden, als REXX-Funktionen deklarieren
RexxFunctionHandler XeliaREXXRegisterFunctions;
RexxFunctionHandler XeliaREXXDeRegisterFunctions;
RexxFunctionHandler XeliaREXXASCIToHTML;

// Tabelle aller REXX-Funktionen zur Registrierung/ Deregistrierung
static PSZ  RexxFunctionTable[] =
{
    "XeliaREXXRegisterFunctions",
    "XeliaREXXDeRegisterFunctions",
    "XeliaREXXASCIToHTML",
};

//Rückgabewerte für die REXX-Funktionen
#define  INVALID_FUNCTION 40      // Fehler ausgeben
#define  VALID_FUNCTION   0       // Fehlerfreie Ausführung der Funktion
```

So kann auch nichts vergessen werden. Den Header selbst binden Sie mit:

```
#include <rexxsaa.h>
```

in die Datei ein. Sie benötigen diesen Header übrigens auch für die REXX-taugliche Version von *xtest.exe*. Der nächste Abschnitt ist ebenfalls obligatorisch. Hier definieren Sie alle Funktionen, die Sie exportieren, als REXX-Routinen. Dazu benutzen Sie die Typendeklaration *RexxFunctionHandler*, welche Sie auch in der Datei *rexxsaa.h* finden:

```
typedef ULONG APIENTRY
        RexxFunctionHandler(PUCHAR,
                            ULONG,
                            PRXSTRING,
                            PSZ,
                            PRXSTRING);
```

Für uns von Interesse sind der zweite, dritte und letzte Parameter. Der zweite Parameter bezeichnet die Anzahl der Argumente, die der Funktion übergeben wurden. Im dritten Parameter finden Sie den Pointer auf einen REXX-String, der die Argumente enthält. Der letzte Parameter ist ebenfalls ein REXX-String und dient zur Definition des Rückgabewertes der Funktion an den Interpreter. Ihre REXX-Routinen haben damit immer den gleichen Funktionskopf. Zur Analyse der einzelnen Parameter kommen wir gleich bei der Betrachtung des C-Codes der DLL.

Anschließend erzeugen wir mit der folgenden Deklaration:

```
static PSZ  RexxFunctionTable[] =
{
   "XeliaREXXRegisterFunctions",
   "XeliaREXXDeRegisterFunctions",
   "XeliaREXXASCIToHTML",
};
```

eine Tabelle, welche die Namen aller REXX-Funktion enthält, die exportiert werden sollen. Wir benötigen diese Tabelle zur bereits erwähnten Registrierung und Deregistrierung der REXX-Funtkionen.

! *Beachten Sie hierbei das Komma nach dem letzten Eintrag in der Tabelle. Es darf nicht fehlen.*

Zum Schluß definieren wir noch zwei Konstanten, die von der Funktion abhängig vom Erfolg oder Mißerfolg eines Codeabschnittes zurückgegeben werden.

Wenden wir uns nun der DEF-Datei zu. *Codebeispiel 5* gibt diese Datei wieder. Da die Schlüsselworte vom ersten Teil dieses Workshops bekannt sind, können wir an dieser Stelle auf eine ausführliche Erklärung verzichten. Beachten Sie lediglich, die EXPORT-Anweisungen so in Ihre eigenen Dateien zu übernehmen, wie wir es das Codebeispiel zeigt. Damit ist sichergestellt, daß die einzelnen Funktionen so exportiert werden,

Codebeispiel 5: Module Definition File der neuen REXX-DLL "xrexx.def"

```
LIBRARY XREXX INITINSTANCE LONGNAMES
PROTMODE
DESCRIPTION 'XREXX'
DATA MULTIPLE NONSHARED
STACKSIZE 32768
EXPORTS
  XELIAREXXASCITOHTML  =XeliaREXXASCIToHTML  @1
  XELIAREXXREGISTERFUNCTIONS =XeliaREXXRegisterFunctions  @2
  XELIAREXXDEREGISTERFUNCTIONS =XeliaREXXDeRegisterFunctions  @3
```

daß man Sie im REXX-Skript problemlos verwenden kann. Die EXPORT-Direktive:

```
XELIAREXXASCITOHTML
                =XeliaREXXASCIToHTML   @1
```

definiert die Funktion *XeliaREXXASCIToHTML* als zu exportierende Methode und weist ihr gleichzeitig ein Pseudonym zu, wobei alle Buchstaben groß geschrieben werden. Damit wird es möglich -wie es unter REXX manchmal gebräuchlich ist-, die Funktion auch komplett in Großbuchstaben in den Programmcode einzufügen. Mit @1 wird der Funktion eine Ordinalnummer zugewiesen. Fügen Sie diese Ordinalnummern stets zu jeder EXPORT-Anweisung hinzu. Damit sind alle Vorbereitungen abgeschlossen. Nun müssen die einzelnen Funktionen nur noch geschrieben werden. Betrachten Sie sich zu den folgenden Erklärungen den C-Code der REXX-DLL, den Sie in *Codebeispiel 6* finden.

Jede DLL, die Sie zur Erweiterung von REXX schreiben, sollte zwei Funktionen enthalten, die das Registrieren und Deregistrieren der neuen Routinen vereinfachen. Sie wissen bereits, daß man eine extern definierte Routine mit *RxFuncAdd* anmelden muß. Ferner muß die neue REXX-DLL auch registriert werden. Vielleicht haben Sie sich schon gefragt, wann diese DLL denn mit *DosLoadModule* geladen wird, so wie Sie es bisher kennen. Diese Aufgabe soll die erste Funktion unserer Bibliothek übernehmen: *XeliaREXXRegisterFunctions*.

REXX-DLLs werden nicht wie üblich über das DOS-API geladen. Vielmehr wird jede Routine der DLL über die API-Funktion *RexxRegisterFunctionDLL* dem Interpreter gemeldet. Für jede Funktion, die man später im REXX-Skript verwenden möchte, muß vom Programmteil, der in C verfaßt wurde, diese API-Funktion aufgerufen werden. Unsere Registrierungsroutine erledigt diese Arbeit in einer *for*-Schleife.

Zunächst überprüfen wir jedoch die der Funktion übergebenen Parameter mit:

```
if (numargs != 0)
  return INVALID_ROUTINE;
```

Wenn Sie noch einmal einen Blick auf unser REXX-Skript werfen, so können Sie ersehen, daß der Funktion *XeliaREXXRegisterFunctions* mehrere Parameter übergeben werden. Mit der oben dargestellten *if*-Abfrage stellen wir sicher, daß dem auch so ist. Derartige Syntax-Überprüfungen müssen Sie in allen REXX-Funktionen Ihrer DLL vornehmen, um einen korrekten Programmablauf zu gewährleisten.

Anschließend bestimmen wir mit Hilfe der zuvor definierten Funktionstabelle die Anzahl der exportierten Funktionen mit:

```
lFuncNum =
      sizeof(RexxFunctionTable)/sizeof(PSZ);
```

und führen schließlich die Registrierung aller Funktionen, die wir in die Tabelle aufgenommen haben, durch:

```
for (iCount = 0; iCount < iFuncNum;iCount++)
  RexxRegisterFunctionDll(RexxFunctionTa-
                    ble[iCount],"XREXX",
                    RexxFunctionTable[iCount]);
```

Als Rückgabewert dient ein NULL-String. Nach erfolgreicher Ausführung dieser Funktion können im REXX-Skript alle neuen REXX-Routinen verwendet werden.

Um für eine korrekte Deregistrierung zu sorgen, dient das Pendant unserer Registrierungsfunktion, das wir *XeliaREXXDeregisterFunctions* genannt haben.

Codebeispiel 6: C-Datei der neuen REXX-DLL "xrexx.c"

```
//--------------------------------------------
// Code für XREXX.DLL
//--------------------------------------------

// Header einbinden
#include "xrexx.h"

//--------------------------------------------
// Exportierte REXX-Funktionen
//--------------------------------------------
//--------------------------------------------
// XeliaREXXRegisterFunctions
//--------------------------------------------
ULONG XeliaREXXRegisterFunctions(CHAR *name, ULONG numargs,
                                 RXSTRING args[],
                                 CHAR *queuename, RXSTRING *retstr)
{
 INT    iFuncNum;    // Anzahl der Funktionen in der Funktionstabelle
 INT    iCount;      // Zählvariable für for-Schleife

 // Anzahl der Argumente übrerprüfen
 if (numargs != 0)
  return INVALID_ROUTINE;

 // Anzahl der in der Funktionstabelle enthaltenen Einträge
 // berechnen
 iFuncNum = sizeof(RexxFunctionTable)/sizeof(PSZ);

 // Alle Funktionen registrieren
 for (iCount = 0; iCount < iFuncNum; iCount++)
  RexxRegisterFunctionDll(RexxFunctionTable[iCount],"XREXX",
                          RexxFunctionTable[iCount]);

 retstr->strlength = 0;       // Nullstring zurückgeben

 return VALID_ROUTINE;        // Funktion fehlerfrei beendet
}

//--------------------------------------------
// XeliaREXXDeRegisterFunctions
//--------------------------------------------
ULONG XeliaREXXDeRegisterFunctions(CHAR *name, ULONG numargs,
                                   RXSTRING args[],
                                   CHAR *queuename, RXSTRING *retstr)
{
 INT    iFuncNum;    // Anzahl der Funktionen in der Funktionstabelle
 INT    iCount;      // Zählvariable für for-Schleife

 // Anzahl der Argumente übrerprüfen
 if (numargs != 0)
  return INVALID_ROUTINE;
```

Codebeispiel 6: C-Datei der neuen REXX-DLL "xrexx.c" (Fortsetzung)

```
// Anzahl der in der Funktionstabelle enthaltenen Einträge
 // berechnen
 iFuncNum = sizeof(RexxFunctionTable)/sizeof(PSZ);

 // Alle Funktionen deregistrieren
 for (iCount = 0; iCount < iFuncNum; iCount++)
   RexxDeregisterFunction(RexxFunctionTable[iCount]);

 retstr->strlength = 0;         // Nullstring zurückgeben

 return VALID_ROUTINE;       // Funktion fehlerfrei beendet
}


//-----------------------------------------
// XeliaREXXASCIToHTML
//-----------------------------------------
ULONG XeliaREXXASCIToHTML(CHAR *name, ULONG numargs,
                                    RXSTRING args[],
                                    CHAR *queuename, RXSTRING *retstr)
{
  PSZ      arg1;      // Erster Parameter
  PSZ      arg2;      // Zweiter Parameter
  FILE *fASCIIfile, *fHTMLfile;  // Dateihandles für die Formatkonvertierung
  char cNewChar;  // Variable für das zeichenweise Auslesen der Datei

  // Überprüfen, ob der REXX-Funktion die richtige Anzahl an Argumenten
  // übergeben wurde. Wenn ja, mit der Bearbeitung fortfahren
  if (numargs > 2)
    return INVALID_ROUTINE;

  // Die Stringpointer des REXX-Strings args[] in die beiden PSZ-Variablen
  // kopieren.
  arg1 = args[0].strptr;             // erster Parameter
  arg2 = args[1].strptr;             // zweiter Parameter

  // Originaldatei zum Lesen öffnen
  // Funktion abbrechen, wenn Datei nicht geöffnet werden konnte
  if (( fASCIIfile = fopen(arg1, "r")) == NULL)
    return INVALID_ROUTINE;
  // HTML-Datei zum Schreiben öffnen
  // Funktion abbrechen, wenn Datei nicht geöffnet werden konnte
  if ((fHTMLfile = fopen(arg2, "w")) == NULL)
    return INVALID_ROUTINE;

  // Originaldatei zeichenweise auslesen, dabei Umlaute sowie das
  // ß konvertieren und in die neu erzeugte HTML-Datei schreiben
  while(!feof( fASCIIfile))
    {
    cNewChar = fgetc(fASCIIfile);
```

Codebeispiel 6: C-Datei der neuen REXX-DLL "xrexx.c" (Fortsetzung)

```
  switch(cNewChar)
    {
    case 'ä':
      fputs("&auml;",fHTMLfile);
      break;
    case 'ü':
      fputs("&uuml;",fHTMLfile);
      break;
    case 'ö':
      fputs("&ouml;",fHTMLfile);
      break;
    case 'Ä':
      fputs("&Auml;",fHTMLfile);
      break;
    case 'Ü':
      fputs("&Uuml;",fHTMLfile);
      break;
    case 'Ö':
      fputs("&Ouml;",fHTMLfile);
      break;
    case 'ß':
      fputs("&szlig;",fHTMLfile);
      break;
    default:
      fputc(cNewChar,fHTMLfile);
      break;
    }   //endswitch
  } //endwhile

  fclose(fASCIIfile);  // Originaldatei schließen
  fclose(fHTMLfile); // HTML-Datei schließen

  retstr->strlength = 0;         // Nullstring zurückgeben

  return VALID_ROUTINE;      // Funktion fehlerfrei beendet
}

// end of file
```

Der Aufbau dieser Funktion ist im wesentlichen der gleiche wie von *XeliaREXXRegisterFunctions*. Lediglich der Anweisungsteil der for-Schleife ist anders, da zum Deregistrieren der Funktionen das REXX-API *RexxDeregisterFunction* verwendet wird. Nachdem wir wieder die Anzahl der in der Tabelle eingetragegenen Funktionen überprüft haben, nehmen wir die Deregistrierung in einer Schleife vor:

```
for (iCount = 0; iCount < iFuncNum;iCount++)
  RexxDeregisterFunction(RexxFunctionTable[iCount]);
```

Diese beiden Funktionen können Sie ohne Änderungen in Ihre eigenen DLLs übernehmen, um Ihre REXX-Routinen schnell und richtig zu registrieren und zu deregistrieren. Wie man beide Funktionen unter REXX aufruf, haben wir bereits weiter oben dargestellt.

Die dritte und letzte Funktion der neuen REXX-DLL nimmt die Konvertierung des ASCII-Textes in das HTML-Format vor. Um den Code einfach zu halten, beschränken wir uns wie im REXX-Beispiel wieder nur auf die Umsetzung der Umlaute und des ß. Die Technik der Formatumsetzung wurde bereits beschrieben. Wir richten daher unser Augenmerk darauf, wie die der Funktion übergebenen Parameter im Programmcode verarbeitet werden.

Wir übergeben der Funktion zwei Parameter, nämlich den Namen der Originaldatei und den des zu erzeugenden HTML-Dokumentes. Betrachten wir dazu den Funktionskopf etwas genauer:

```
ULONG XeliaREXXASCIToHTML(CHAR
                          *name,
                          ULONG numargs,
                          RXSTRING args[],
                          CHAR *queuename,
                          RXSTRING *retstr)
```

Im Parameter *numargs* finden wir die Anzahl der übergebenen Parameter wieder. Der folgende Parameter *args[]* stellt ein Array von REXX-Strings dar, welche die Argumente enthalten, die an unsere Funktion übergeben wurden. Um mit ihnen komfortabler arbeiten zu können, speichern wir die Pointer auf die beiden Argumente in zwei PSZ-Variablen:

```
arg1 = args[0].strptr;
arg2 = args[1].strptr;
```

arg1 enthält damit den Zeiger auf den ersten REXX-String, *arg2* weist auf den zweiten. Anschließend können wir mit den Zeigern wie gewohnt arbeiten, z.B. eine Datei öffnen. Übergeben Sie einer Funktion im REXX-Programm einen numerischen Wert, müssen Sie diesen im Funktionscode zur Weiterverarbeitung zunächst in eine Ganz- oder Gleitkommazahl konvertieren. Dazu können Sie die C-Bibliotheksfunktionen verwenden. Nehmen wir z.B. an, Sie übergeben einer REXX-Funktion eine Ganzzahl und eine Gleitkommazahl, dann könnte der Programmcode zur Verwendung dieser Parameter so aussehen:

```
// Variablendeklarationen
double d;
long l;

// Typkonvertierung der Parameter
d = atof (args[0].strptr);
l = atol (args[0].strptr);
```

Anschließend läßt sich mit den beiden Werten rechnen. Das Ergebnis kann man letztlich in eine Datei schreiben. Hat man es nur mit einem Einzelergebnis zu tun, übergibt man am Ende der Funktion keinen Leerstring in der Variablen *retstr*, sondern das Ergebnis, das evtl. erst wieder in einen String konvertiert werden muß, ehe es zurückgegeben wird. Dazu noch ein Beispiel: Die oben genannte REXX-Funktion gibt nach ihren Berechnungen eine Fließkommazahl aus. Der Code dazu sähe wie folgt aus:

```
sprintf(retstr->strptr, "%f", d);
retstr->strlength = strlen(retstr->strptr);
```

Verwenden Sie weitere API-Funktionen und möchten einen API-Rückgabecode ausgeben, lautete der Code wie folgt:

```
sprintf(retstr->strptr, "%d", rc);
retstr->strlength = strlen(retstr->strptr);
```

wenn *rc* die Variable ist, die den Returncode enthält. Im REXX-Programm können Sie den Rückgabewert wie folgt verarbeiten:

```
Result = RexxFunktion ('1.125','25')
```

Mit der Variablen *Result* können Sie dann direkt im REXX-Programm weiterrechnen.

Mit diesen abschließenden Hinweisen wollen wir den zweiten Teil unseres Workshops beenden. Zuvor geben wir noch einen Ausblick auf den nächsten und letzten Teil dieses Artikels.

Nachteile des Beispielprogramms

Sie bemerken vielleicht, daß unser Beispielprogramm das Benutzerinterface stillegt, wenn die REXX-Routine ausgeführt wird, gleich ob sich der Code zur Dateiformatkonvertierung in einer C-DLL oder direkt in der REXX-Datei befindet (wenngleich im ersten Fall die Ausführungsgeschwindigkeit sehr gesteigert wurde). Das liegt zum einen daran, daß ein zeichenweises Auslesen einer Datei ein zeitaufwendiger Prozeß ist. Man könnte den Algorithmus noch weitgehend verbessern, um die Datei zum einen vollständig (nicht nur die Umlaute), zum anderen schneller zu konvertieren. Dennoch wird die graphische Oberfläche eine ganze Zeit lang nicht auf Benutzereingaben reagieren. Dieses Verhalten kennen Sie sicher von vielen unterschiedlichen Programmen. Bedingt wird es dadruch, daß der PM ein nachrichtenbasiertes System ist: Alle Benutzereingaben werden in einer einzigen Nachrichtenschlange, der **System Queue**, gespeichert. Wird eine Nachricht an eine Applikation gesendet, holt sie sich diese von der *System Queue* und bearbeitet sie. Die Kontrolle wird aber erst danach an das System zurückgegeben, so daß die nächste Nachricht zur Bearbeitung von der Systemwarteschlange erst dann geholt werden kann, wenn die vorherige Nachricht vollständig bearbeitet ist. Ein Nachrichtensystem, das mit mehreren *Systemqueues* arbeitet, wird diesen Effekt nicht aufweisen (z.B. Windows NT). Daher wünschen sich viele OS/2-Anwender auch eine *Multiple System Queue* für den PM. Dabei wird aber oftmals nicht beachtet, daß mehrere Warteschlangen den Verwaltungsoverhead des Betriebssystems vergrößern, was der Gesamtperformance auf gar keinen Fall zugute kommen kann. Ferner gibt es viele NT-Anwendungen, die mit diesem Feature auf eine solche Weise umgehen, daß die ohnehin OS/2 unterlegene Systemstabilität noch weiter herabgesetzt wird.

Unter OS/2 und dem PM benutzt man daher anstelle mehrerer Warteschlangen **Threads**, um Funktionen, die länger als 1/10 Sekunde zur Bearbeitung benötigen, quasiparallel zum übrigen Programmcode auszuführen: Nach dem Starten des *Threads* wird die Kontrolle vom Programm sofort wieder an den PM übergeben, wodurch das Benutzerinterface weiterhin aktiv bleibt. Die zeitaufwendige Funktion läuft dann in einem eigenen Thread, d.h. ihr wird Prozessorzeit vom System zugeteilt. Damit lassen sich mehrere Aufgaben gleichzeitig innerhalb eines Prozesses ausführen. Leider ist das Konzept des **Multithreading** in den wenigsten OS/2-Anwendungen konsequent implementiert. Teilweise fehlt es ganz; manchmal ist es unsauber in das Programm integriert, so daß es dessen Stabilität beeinträchtigt.

Wir werden unser Beispielprogramm daher sinnvoll und OS/2-konform durch *Multithreading* erweitern. Dabei verfahren wir jedoch so, die schwierigste Arbeit nur einmal zu tun. Ziel soll es somit sein, eine Architektur zu entwickeln, die *Multithreading* -egal für welches Programm- einfach und sicher anzuwenden macht und alle Möglichkeiten dieses Features ausnutzt. Dies führt uns zum dritten Anwendungsbereich von DLLs: Zu *Subsystemen.* Im dritten und letzten Teil dieses Workshops werden wir daher die Entwicklung eines *Multithreading-Subsystems* betrachten, welches den Nachteil unseres Beispielprogrammes mit einigen Funktionsaufrufen beseitigen wird. ❑

Netzwerke mit OS/2, Teil 1

OS/2 bietet seit der Version 3 mit *Warp Connect* viele und v.a. einfache Möglichkeiten der Vernetzung mehrerer Rechner. Mit *Warp 4* wurde die Netzwerkunterstützung in den Lieferumfang des Basisproduktes integriert, so daß zu einem akzeptablen Preis auch dem Endanwender alle Möglichkeiten zum Aufbau eines Netzwerkes offenstehen. Wenn man bedenkt, daß man außer dem Betriebssystem keine weiteren Zusatzprodukte benötigt, um ein Netzwerk aufzubauen, wird der besondere Vorteil von OS/2 hinsichtlich des *Network Computing* klar, auch im Vergleich mit anderen Computersystemen.

Nun schrecken viele Anwender vor der Planung und Realisierung eines eigenen Netzwerkes zurück. Doch so schwierig ist es gar nicht, und selbst für den Endanwender kann ein Netzwerk von Interesse sein. Wer neben seinem Einzelplatzsytem noch ein Notebook sein eigen nennt, wird von ganz alleine nach einer Möglichkeit suchen, Daten von dem einen auf das andere Sytem zu übertragen. Hier mit Disketten zu arbeiten, wird mit der Zeit zu einer nervenaufreibenden und bestimmt nicht mehr zeitgemäßen Arbeit.

Dieser Netzwerkworkshop zeigt Ihnen, wie Sie Ihre OS/2-Rechner miteinander vernetzen, egal ob es sich nur um zwei Systeme handelt, die miteinander verbunden werden sollen, oder um mehrere dutzend oder gar hundert Computer. Der erste Teil des Workshops vermittelt das Basiswissen, über das Sie verfügen sollten, bevor Sie zur Tat schreiten. Im zweiten Teil stellen wir Ihnen die zur Vernetzung nötige Hardware vor und vergleichen verschiedene Produkte miteinander. Außerdem zeigen wir Ihnen, wie Sie das Netzwerk, das sich für Ihre Belange am besten eignet, planen und die dazu notwendigen Hardwarekomponenten installieren. Der dritte Teil des Workshops beschäftigt sich dann mit der Installation der erforderlichen Software sowie der Konfiguration des Netzes und dessen Inbetriebnahme. Hier stellen wir Ihnen die interessantesten mit OS/2 aufbaubaren Netzwerktypen vor und geben Tips und Tricks zum Aufbau unterschiedlicher Netze.

Einige der innerhalb des Workshops vorgestellten Verbindungsmöglichkeiten zwischen zwei oder mehreren Rechnern sind eher für den Endanwender geeignet; für professionelle Aufgaben stellen wir Ihnen die Realisierung eines TCP/IP-Netzes vor. Wir verzichten jedoch auf die genaue Beschreibung typischer Client-Server-Architkturen und damit auch auf die Betrachtung des *Warp Servers*, wie man ihn installiert und einrichtet. Grundsätzlich kann man mit einem normalen *Warp 4* genauso komplexe und leistungsfähige (wenn nicht sogar leistungsfähigere aber auf jeden Fall kostengünstigere) Netzwerke aufbauen wie mit dem Warp-Server, an den mehrere Clients angeschlossen sind. Mit TCP/IP als Netzwerkprotokoll ist der Aufbau selbst großer Netze gar kein Problem, sehr viel flexibler als andere Architekturen und läßt daneben auch die einfachere Eingliederung Nicht-OS/2-Rechner, etwa einer Unix-Maschine zu. Selbst einen *Macintosh* kann man nachträglich recht einfach in ein TCP/IP-Netz 'einklinken'. Wie man heterogene Netze aufbaut, zeigen wir Ihnen im dritten Teil dieses Workshops. Da wir uns schwerpunktmäßig mit TCP/IP-Netzen beschäftigen werden, wird dieses Protokoll auch im Mittelpunkt des einführenden Teils stehen.

Was ist ein LAN?

Die Abkürzung für LAN steht für *Local Area Network* oder zu deutsch: *Lokales*

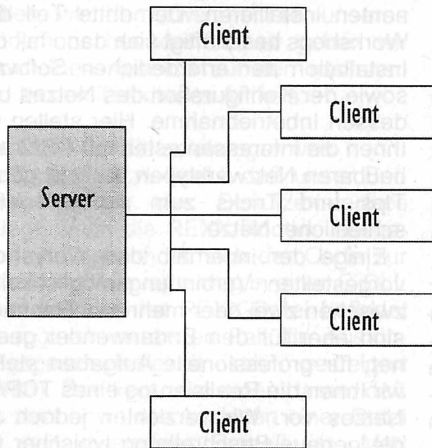


Abb. 1: Schematische Darstellung eines Client/ Server-Netzwerkes

Netzwerk. Der Gegensatz dazu ist das *WAN (Wide Area Network)*. Die Begrifflichkeit erklärt zugleich den Einsatzzweck: Ein *LAN* ist räumlich auf ein Zimmer, ein Stockwerk, ein Gebäude oder einen Gebäudekomplex beschränkt, während ein *WAN* ein Netzwerk darstellt, bei dem die Rechner über weite Distanzen miteinander verbunden sind, etwa über mehrere Städte hinweg. Dementsprechend unterschiedlich sind natürlich auch die Übertragungswege, v.a. was die Methoden der Signalübertragung von Rechner zu Rechner und die davon abhängige maximal zulässige Länge der einzelnen Verbindungsleitungen angeht.

Per definitionem handelt es sich bei einem *LAN* um eine Anzahl miteinander verknüpfter Computer, die sowohl unabhängig als auch gemeinsam betrieben werden können. Die Anzahl der miteinander verknüpften Rechner kann dabei zwischen zwei oder mehreren hundert Maschinen liegen. Prinzipiell ist also der Zusammenschluß eines Notebooks und eines stationären Systems ebenfalls ein *LAN* - solange die beiden Rechner miteinander verbunden sind.

Der Zweck eines *LANs* ist neben der Datenspeicherung auch die Bereitsstellung dieser Daten an alle Benutzer, die an das *LAN* angeschlossen sind. Das gilt auch für an einen der Netzwerkrechner angeschlossene Peripheriegeräte, etwa einen Drucker, der dann von allen *LAN*-Computern genutzt werden kann. Kurz gesagt ist die Aufgabe eines *LAN*, Ressourcen zu verwalten und zu verteilen. In einem Unternehmen kann man so verschiedene Abteilungen miteinander verbinden. Für Privatanwender lassen sich z.B. Mobilgerät und Schreibtischcomputer zusammenschließen, um Daten schnell zu übertragen; auch kann man sich via Modem von unterwegs mit dem Notebook auf dem heimischen Rechner einwählen, um rasch wichtige Daten zu transferieren. Mit der oben angegebenen Einsatzmöglichkeit könnte man auch eine unterwegs erstellte Datei auf einem hochwetigen Drucker zu Hause ausgeben lassen. Die Anwendungsfälle sind also mannigfaltig, und der Phantasie sind kaum Grenzen gesetzt.

Vergleichbar mit einem *LAN* ist ein Großrechner, an den mehrere Terminals (Arbeitsplätze) angeschlossen sind. Alle Benutzer können sich die Rechenleistung dieses Rechners teilen. Je mehr Benutzer man hat, desto weniger Ressourcen stehen natürlich zu Verfügung. Beim *LAN* können es mehrere Rechner sein, die auch mehreren Benutzern zur Verfügung stehen. Ein weitaus wichtigerer Vorteil aber besteht in der größeren Sicherheit gegen den Ausfall eines Rechners im Netz. Während ein Hauptrechner bei einem Systemausfall

keinem der angeschlossenen Benutzer eine Fortsetzung der Arbeit gestattet, legt das Versagen einer Maschine im *LAN* das Netz nicht lahm. Zudem kann ein zweiter Rechner für den ausgefallenen Computer »einspringen«. Auch ist ein *LAN* in seiner Kapazität einfach erweiterbar. Das alles sind Gründe, wieso sich Computernetze gegenüber den Lösungen mit Großrechnern und Terminals durchgesetzt haben und auch weiterhin durchsetzen werden.

Client/Server und Peer-to-Peer
Grundsätzlich unterscheidet man zwischen zwei verschiedenen Netzwerkarchitekturen: Den *Client/Server*- und *Peer-to-Peer*-Netzwerken.

Im ersten Fall richtet man Maschinen ein, die als *Server* (Anbieter) dienen, welche den *Clients* (Nutzern) Daten oder Ressourcen zur Verfügung stellen (s. *Abb. 1*). Ein *Server* ist dabei ein dediziertes System, das neben der Verwaltung der einzelnen *Clients* lediglich das Angebot an *Peripherie* (z.B. Druckern) und Daten übernimmt, während die einzelnen Clients nichts anderes tun, als von diesem Angebot Gebrauch zu machen. Jede Kommunikation der *Clients* untereinander läuft dabei über den *Server*. Fällt diese Maschine aus, und gibt es kein anderes System, das ihre Aufgabe übernimmt, so liegt auch das Netzwerk lahm. Dafür sind *Client/Server*-Architekturen zumeist leistungsfähiger als ein *Peer-to-Peer*-Netzwerk, was logisch erscheint, da für bestimmte Aufgaben im Netz speziell ausgebaute Rechner zur Verfügung stehen. Außerdem verläuft die Verwaltung aller Arbeitsstationen über eine Maschine, was den administrativen Aufwand erheblich vermindert. Keiner der einzelnen *Clients* muß die Adresse der anderen *Clients* im Netz kennen, lediglich die des *Servers*. Daraus ergeben sich gleichzeitig bessere Implementierungsmöglichkeiten von Sicherheitskonzepten.

Ein *Peer-to-Peer*-Netz kennt eine derartige Hierarchie nicht (s. *Abb. 2*). Alle Rechner sind zu einem Verbund zusammengeschlossen, in dem jedes System sowohl die Aufgaben eines *Servers* als auch die des *Clients* übernehmen kann (wenn auch nicht muß). Die einzelnen Rechner können ohne Hindernis miteinander kommunizieren, vorausgesetzt, System A kennt System B, B kennt C usw. Eine zentrale Verwaltung der einzelnen Netzwerkteilnehmer gibt es nicht. Sollte eine der Maschinen ausfallen, so bleibt das Netzwerk in aller Regel davon unberührt.

Beide Architekturen haben Vor- und Nachteile. Für die meisten Aufgaben ist jedoch ein *Peer*-Netz geeigneter als eine auf der *Client/Server*-Architektur basierenden Lösung. Daneben existiert keine so klare Trennung zwischen den beiden Architekturen, wie man annehmen könnte. Vielerlei Merkmale der einen sind in die andere übergegangen. Zusammen-

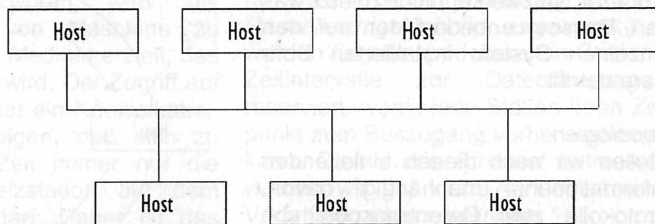


Abb. 2: Beispiel für ein Peer-to-Peer-Netzwerk. Die hier dargestellte Topologie dient nur der Anschauung und ist nicht etwa zwingend vorgeschrieben.

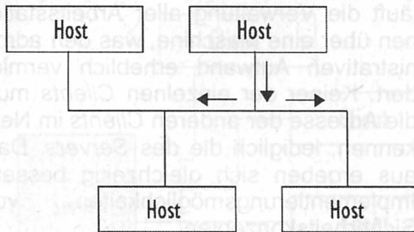


Abb. 3: Bussystem

fassend läßt sich sagen, daß grundsätzlich kein Argument existiert, das ohne Ausnahme für den notwendigen Einsatz einer typischen *Client/Server*-Lösung spricht. Selbst wenn man es mit einem großen Netzwerk (also mit mehreren hundert *Hosts*) zu tun hat, so ist ein *Peer*-Netzwerk immer noch eine akzeptable Lösung. Man kann innerhalb eines *Peers* einzelne Rechner auch für Serverzwecke dedizieren. Das können z.B. Druck- oder Dateiserver sein, die zwar theoretisch auch als Clients genutzt werden können, praktisch aber nicht genutzt werden. In einem TCP/IP-Netz bedient man sich häufig dieser Technik. Generell sind TCP/IP-Netze sehr leistungsfähig. Das normale *OS/2 Peer*, das wir innerhalb dieses Workshops ebenfalls besprechen, zeigt seine Schwächen gegenüber dem weitaus flexibleren TCP/IP v.a. beim Aufbau größerer Netzwerke, nicht zuletzt was den Ressourcenbedarf der auf den einzelnen System installierten Software betrifft.

Topologien

Stellen wir nach diesen einleitenden Informationen, unabhängig davon, Protokolle zum Datentransport von einem Rechner zum anderen zu betrachten, einige Überlegungen zum physikalischen Aufbau von Netzwerken an.

Die Art und Weise, mit der einzelne Stationen in einem Rechnerverbund miteinander gekoppelt sind, bezeichnet man als **Topologie**.

Es existieren vielfältige Möglichkeiten, mehrere Maschinen miteinander zu verbinden. Grundformen sind Bussystem (*Abb. 3*), Ringsystem (*Abb. 4*), Sternsystem (*Abb. 5*) und das Maschennetz. Zusätzlich existieren die unterschiedlichsten Mischformen, die wir jedoch nicht betrachten. Wir beschränken uns auf eine nähere Untersuchung der bekanntesten Vertreter des Bus- und des Ringsystems, dem *Ethernet* und dem *Token-Ring*, da beide Systeme in der Praxis sehr oft anzutreffen sind und für unsere Bedürfnisse in Frage kommen. Den grundsätzlichen Aufbau eines Bus-, Ring- und Sternsystems zeigen die schematischen *Abbildungen 3-5*. Wir sehen von einer Besprechung des Ringsystems ab, da es zunehmend an Bedeutung verliert.

Für den Endanwender bietet das Bussystem meist die am besten geeignete Topologie, da sich ein *Ethernet*-System sehr einfach und schnell aufbauen läßt. Die Leitungsführung muß nicht geschlossen sein, und zusätzliche Maschinen lassen sich einfach in das Netz integrieren. Vor allem für kleine

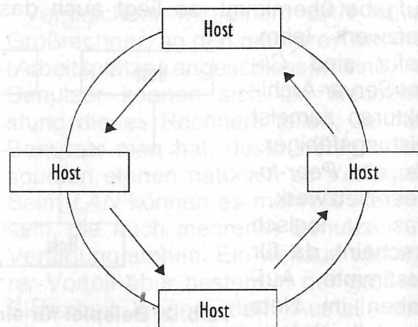


Abb. 4: Token Ring

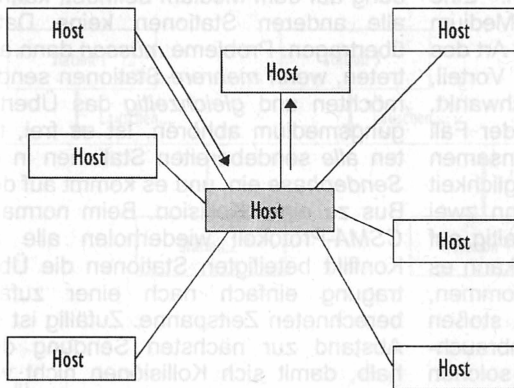


Abb. 5: Sternsystem. Diese Topologie ist relativ unbedeutsam, da die Störanfälligkeit zu hoch und der Datendurchsatz im allgemeinen zu gering ist.

Netze bietet sich dieser Verbundtyp mehr an als ein Ringsystem. Generell muß ein Bussystem aber nicht die beste Wahl sein. Welche der beiden Topologien man wählt, wird letztlich vom Einsatzzweck der einzelnen Stationen abhängen, von der Größe des Netzes und den Anwendungen, die auf den einzelnen Maschinen laufen. Im zweiten Teil dieses Workshops werden wir uns detailliert mit der Planung des richtigen Netzes beschäftigen.

Das Ethernet-Bussystem

Bei einem Bussystem wird die Signalübertragung von Maschine zu Maschine mit einem Medium erzielt, das gemeinsam benutzt wird. Der Zugriff auf dieses Medium, meist ein Koaxialkabel, muß dabei so erfolgen, daß sich zu einer bestimmten Zeit immer nur die Sendung *einer* Netzstation auf dem Medium befinden darf. Daher ist das bedeutendste Problem dieser Systeme den Zugang der einzelnen Stationen auf den Bus zu regeln. Es gibt dazu unterschiedliche Verfahren. Wir werden eine der *Random-Access-Methoden*, die dem *Ethernet* zugrunde liegt, betrachten.

Wie beim Ringsystem, das wir im nächsten Abschnitt besprechen, bietet auch das Bussystem den Vorteil, sehr einfach aufgebaut zu sein. Neue Stationen lassen sich simpel in das Netz integrieren oder auch entfernen, was die Modifikationskosten niedrig hält. Der Ausfall einer Station bringt nur die Unerreichbarkeit dieser Maschine mit sich, d.h. man kann zu ihr nicht senden und nicht von ihr empfangen. Die Gesamtfunktionalität des Systems bleibt dabei allerdings unberührt, womit ein Bussystem, im Gegensatz zum Ringsystem, weniger störanfällig ist.

Hinsichtlich der Auswahltechniken, die nach einer Signalübertragung die nächste sendeberechtigte Station festlegen, unterscheidet man zwischen den bereits erwähnten *Random-Access-Methoden* und den *Reservierungsverfahren*. Im ersten Fall kann der Zeitpunkt des Zugangs einer bestimmten Station zum Bus von dieser Station nicht vorherbestimmt werden. Wird eine *Reservierungstechnik* angewandt, so werden für alle sendebereiten Stationen Zeitintervalle zur Datenübertragung reserviert, womit jede Station ihren Zeitpunkt zum Buszugang vorherbestimmen kann. Da wir das *Ethernet* betrachten, werden wir uns dem *Random-Access-Verfahren* näher zuwenden.

Random-Access-Methoden

Bei diesem Verfahren haben alle Stationen zu *jeder* Zeit Zugriff auf den Bus,

also das Übertragungsmedium. Eine Vorschrift, die den Zugang zum Medium vorschreibt, existiert nicht. Diese Art des Buszugriffs ist immer dann von Vorteil, wenn die Netzaktivität stark schwankt, wie es bei den meisten LANs der Fall ist. Durch den jederzeit gemeinsamen Zugriff ergibt sich jedoch die Möglichkeit zu Übertragungskonflikten: Wenn zwei oder mehrere Stationen gleichzeitig auf das Medium zugreifen können, kann es zu Kollisionen auf dem Bus kommen, d.h. zwei oder mehr Signale stoßen zusammen und werden dabei unbrauchbar oder ganz zerstört. In einem solchen Falle muß die sendende Station die Übertragung wiederholen. Das bedeutet, daß Kollisionen den Datendurchsatz auf dem Bus stark reduzieren, was zu höheren Wartezeiten führt. Dabei steigt die Wartezeit mit der Anzahl der Stationen, die auf den Bus zugreifen wollen. Um diesen Nachteil zu beseitigen, kann man entweder versuchen, die Anzahl an Kollisionen zu verringern; oder man bemüht sich, die durch die Kollisionen bedingte Wartezeit zu verkürzen. Dem *Ethernet* liegt das sogenannte *CSMA-CD*-Verfahren zugrunde (***C**arrier **S**ense **M**ultiple **A**ccess with **C**ollision **D**etection*), welches die zweite der oben genannten Methoden verfolgt. Dieses Protokoll werden wir im folgenden genauer untersuchen.

Den Zugriff auf den Bus kann man bei *CSMA-CD* in drei Phasen einteilen: Die *Lauschphase*, die *Sendephase* und die *Kollisionsphase*. In der *Lauschphase* »hört« eine Station, die Daten senden möchte, das Übertragungsmedium ab und erhält *nur dann* einen Zugang zum Bus, wenn das Medium *frei* ist. In diesem Falle beginnt die Station mit der *Sendephase*, also dem Übertragen der Daten, die in beiden Richtungen auf dem Bus entlanglaufen und alle Stationen erreichen. Während sich eine Sendung auf dem Medium befindet, können alle anderen Stationen keine Daten übertragen. Probleme *müssen* dann auftreten, wenn *mehrere* Stationen senden möchten und *gleichzeitig* das Übertragungsmedium abhören. Ist es frei, treten *alle* sendebereiten Stationen in die *Sendephase* ein, und es kommt auf dem Bus zu einer Kollision. Beim normalen *CSMA*-Protokoll wiederholen alle am Konflikt beteiligten Stationen die Übertragung einfach nach einer zufällig berechneten Zeitspanne. Zufällig ist der Abstand zur nächsten Sendung deshalb, damit sich Kollisionen nicht wiederholen.

Bei *CSMA-CD* kann eine Station ihre eigene Sendung »mithören«, d.h. sie merkt nach einer sehr kurzen Abhörzeit, daß ihre Sendung mit einer anderen kollidiert ist. Tritt dieser Fall ein, brechen die an der Kollision beteiligten Stationen ihre Sendungen sofort ab. Die Zeit, die bis zum Erkennen der Kollision verstreicht, kann dabei für die einzelnen Stationen unterschiedlich groß sein. Um jedoch die Kollisionsphase möglichst klein zu halten, sendet die erste Station, die eine Kollision ihrer Sendung mit der einer anderen Maschine im Netz bemerkt, ein spezielles Störsignal. Sobald dieses Signal auf dem Medium erscheint, ist es auch den anderen Stationen möglich, sich im Gange befindende Sendungen unverzüglich abzubrechen, selbst wenn diese Maschinen eine Kollision ihrer Sendungen noch nicht registriert haben. Dadurch reduziert sich die Dauer der möglichen *Kollisionsphasen* enorm, womit der Durchsatz auf dem Bus steigt, da er nicht mehr mit so vielen, unbrauchbar gewordenen Sendungen, die miteinander kollidiert sind, belastet wird. Damit sinkt auch die Wartezeit erheblich. Nach der Kollisionsphase wiederholen die sendebereiten Stationen ihre Übertragungen

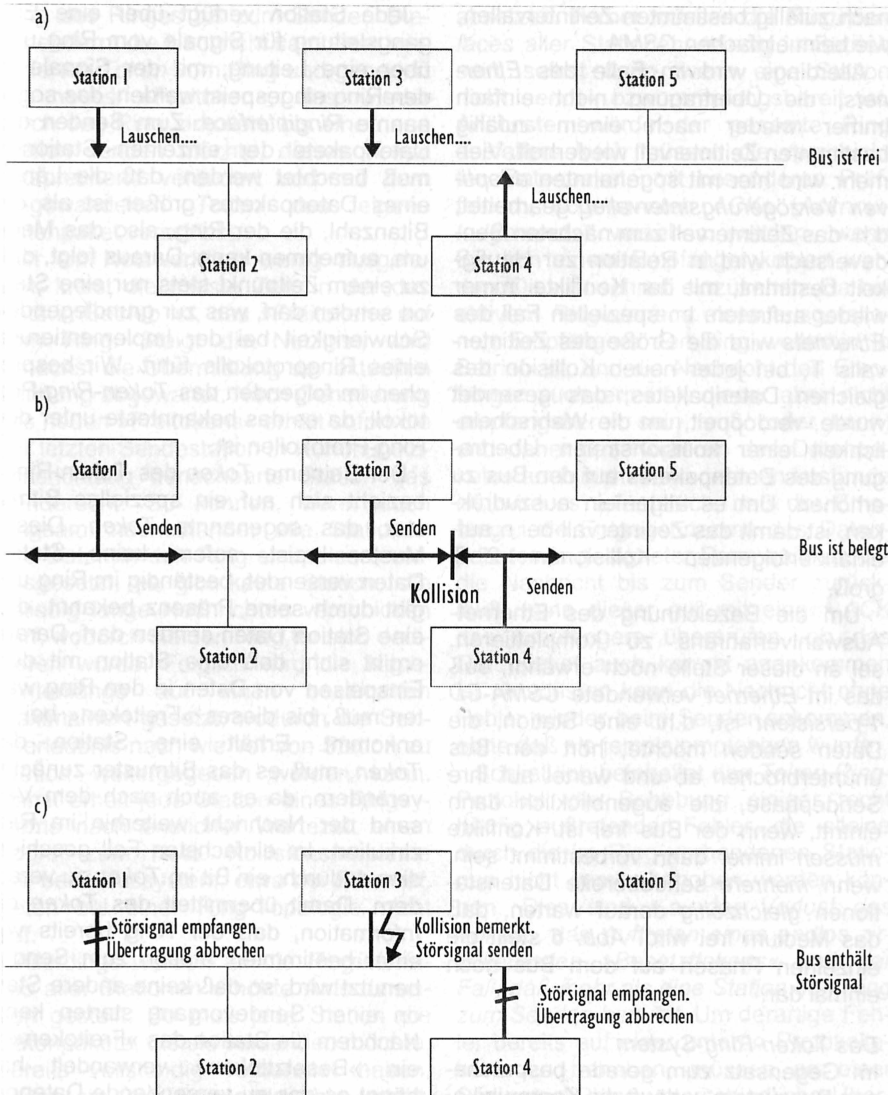


Abb. 6: Das CSMA/CD-Verfahren bei *Ethernet*. Die Stationen 1, 3 und 4 sind sendebereit und hören den Bus ab. Da er frei ist, treten alle drei Stationen in die Sendephase ein. Die Datenpakete der Stationen 3 und 4 kollidieren auf dem Bus. Die Station 3 registriert die Kollision als erste, bricht ihre Übertragung ab und sendet ein spezielles Störsignal aus. Die Stationen 1 und 4 empfangen dieses Signal und brechen ihre Übertragungen auf den Bus ebenfalls augenblicklich ab, selbst wenn Sie die Kollision ihrer Datenpakete noch nicht registriert haben.

nach zufällig bestimmten Zeitintervallen, wie beim einfachen *CSMA*.

Allerdings wird im Falle des *Ethernets*, die Übertragung nicht einfach immer wieder nach einem zufällig bestimmten Zeitintervall wiederholt. Vielmehr wird hier mit sogenannten *adaptiven Verzögerungsintervallen* gearbeitet, d.h. das Zeitintervall zum nächsten Sendeversuch wird in Relation zur Häufigkeit bestimmt, mit der Konflikte immer wieder auftreten. Im speziellen Fall des *Ethernets* wird die Größe des Zeitintervalls T, bei jeder neuen Kollision des gleichen Datenpaketes, das gesendet wurde, verdoppelt, um die Wahrscheinlichkeit einer kollisionsfreien Übertragung des Datenpaketes auf den Bus zu erhöhen. Um es allgemein auszudrükken, ist damit das Zeitintervall bei n aufeinanderfolgenden Kollisionen $2^n \cdot T$ groß.

Um die Bezeichnung des Ethernet-Auswahlverfahrens zu komplettieren, sei an dieser Stelle noch erwähnt, daß das im *Ethernet* verwendete *CSMA-CD 1-persistent* ist, d.h. eine Station, die Daten senden möchte, hört den Bus ununterbrochen ab und wartet auf ihre Sendephase, die augenblicklich dann eintritt, wenn der Bus frei ist. Konflikte *müssen* immer dann vorbestimmt sein, wenn *mehrere* sendebereite Datenstationen *gleichzeitig* darauf warten, daß das Medium frei wird. *Abb. 6* stellt die einzelnen Phasen auf dem Bus noch einmal dar.

Das Token-Ring-System

Im Gegensatz zum gerade besprochenen Bussystem verläuft die Kommunikation auf einem Ring, wie ihn *Abb. 4* zeigt, unidirektional, d.h. immer nur in eine Richtung von Station zu Station, der *Kommunikationsrichtung* (während die Signalausbreitung beim Bussystem immer in beide Richtungen verläuft.

Jede Station verfügt über eine Eingangsleitung für Signale vom Ring und über eine Leitung, mit der Signale in den Ring eingespeist werden, das sogenannte *Ringinterface*. Zum Senden der Datenpakete der einzelnen Stationen muß beachtet werden, daß die Länge eines Datenpaketes größer ist als die Bitanzahl, die der Ring, also das Medium, aufnehmen kann. Daraus folgt, daß zu einem Zeitpunkt stets nur eine Station senden darf, was zur grundlegenden Schwierigkeit bei der Implementierung eines Ringprotokolls führt. Wir besprechen im folgenden das *Token-Ring*-Protokoll, da es das bekannteste unter den Ring-Protokollen ist.

Der Beiname *Token* des Token-Rings bezieht sich auf ein spezielles Bitmuster, das sogenannte *Token*. Dieses Muster kreist, sofern keine Station Daten versendet, beständig im Ring und gibt durch seine Präsenz bekannt, daß eine Station Daten senden darf. Daraus ergibt sich, daß eine Station mit dem Einspeisen von Daten in den Ring warten muß, bis dieses »Freitoken« bei ihr ankommt. Erhält eine Station das *Token*, muß es das Bitmuster zunächst verändern, da es auch nach dem Versand der Nachricht weiterhin im Ring zirkuliert. Im einfachsten Fall geschieht dies dadurch, ein Bit im *Token* zu verändern. Damit übermittelt das *Token* die Information, daß der Ring bereits von einer bestimmten Station zum Senden benutzt wird, so daß keine andere Station einen Sendevorgang starten kann. Nachdem die Station das »Freitoken« in ein »Besetzttoken« verwandelt hat, hängt es das zu versendende Datenpaket diesem *Token* an. Daraufhin läuft das *Token-Datenpaket-Paar* durch den Ring. Der Empfänger des Datenpaketes erkennt die Sendung und fertigt sich eine Kopie an. Anschließend vollendet die Nachricht die begonnene Umkrei-

sung des Ringes bis zum Sender. Dieser entfernt die Nachricht samt dem veränderten *Token* vom Ring und generiert ein neues »Freitoken«, das wieder durch den Ring zirkuliert, bis eine neue Station senden möchte, dieses *Token* entsprechend verändert und mit dem umgewandelten *Token* das eigene Datenpaket verschickt.

Ist ein Netzwerkring wenig ausgelastet, kreist die meiste Zeit, in der das Netz in Betrieb ist, das »Freitoken« auf dem Ring. Steigt die Netzlast, wird zunächst die Übermittlung der aktuellen Sendung abgewartet. Nach Generierung des neuen »Freitokens« erhält sofort die der letzten Sendestation in Kommunikationsrichtung benachbarte Station das Senderecht. So können, wenn auch langsam, nacheinander alle Stationen ihre Daten in den Ring einspeisen, vorausgesetzt, es gibt keine Station, die beliebig lange Nachrichten verschicken darf, wodurch sie den Ring für sich dedizieren würde. Folglich wird die Nachrichtenlänge auf einen bestimmten Maximalwert gesetzt, wodurch die Sendeerlaubnis nach wie vor von Station zu Station weitergegeben werden kann. Damit erhält jede Station eines Ringsystems nach endlicher Wartezeit einen Zugriff zum Netz. Kollisionsprobleme wie beim Bussystem, etwa bei *Ethernet*, treten bei einer Ring-Topologie nicht auf.

Allerdings werden an die *Ringinterfaces* aller Stationen erhöhte Anforderungen gestellt. So muß jede Station die Datenpakete, die sie versenden möchte, bereits vollständig vorbereitet haben, bevor sie die Sendeerlaubnis erhält. Würde eine Station ihre Datenpakete erst dann generieren, wenn sie das Senderecht bekommt, was sehr plötzlich geschehen kann, würden unangenehme Verzögerungen auftreten, die den Gesamtdurchsatz des Netzes herabsetzen. Ferner *müssen* die *Ringinterfaces* aller Stationen *ständig* in Betrieb sein, selbst dann, wenn eine Station nicht sende- bzw. empfangsbereit ist. Ansonsten würde der gesamte Ring ausfallen. Auch müssen die verschickten Datenpakete mit besonderen Prüffeldern, sogenannten *ACKs* (*Acknowledgements*) versehen werden, wenn Sie von einem Empfänger kopiert werden. Dies ist schnell einzusehen: Da bei aktivem *Ringinterface* eines ausgefallenen Empfängers der Ring weiterhin in Betrieb ist, und die Nachricht den Empfänger auch erreicht; dieser aber nicht empfangsbereit sein muß (oder kann) und daher das Kopieren des Datenpaketes auch nicht vorzunehmen imstande ist, wird es erforderlich, daß der Empfänger die Entgegennahme des Datenpaketes in geeigneter Form quittiert. Da die Nachricht bis zum Sender zurückläuft, kann dieser nur mit einem *ACK* des Empfängers überprüfen, ob das Datenpaket auch korrekt angekommen ist. Ansonsten kann die Nachricht ohne Fehler wieder beim Sender ankommen, ohne daß sie jemals empfangen wurde.

Schließlich beinhaltet das *Token-Ring*-Protokoll die Behebung einiger recht häufig auftretender Fehler, die alleine durch die im Ring vorhandenen Stationen nicht immer behoben werden können. Dies sind v.a. der *Verlust des Tokens*, das *Auftreten eines endlos zirkulierenden »Besetzttokens«* und *der Fall, daß mehr als eine Station den Ring zum Senden benutzt.* Um derartige Fehler bereits auf einer unteren Protokollebene zu beheben, müssen an einer Stelle des Rings unterschiedliche Überprüfungen vorgenommen werden.

Sollte ein *Token* verloren gehen, muß geprüft werden, ob in einem für den Ring spezifischen, vorgeschriebenen Zeitintervall das »Freitoken« an einer Stelle des Rings vorbeikommt. Sollte

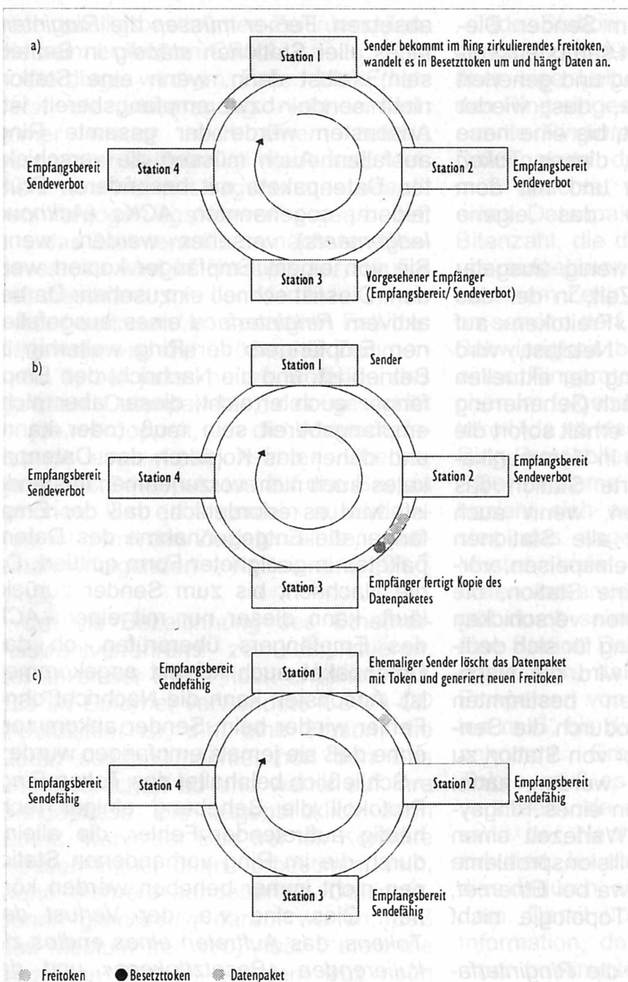


Abb. 7: Ablauf der Datenkommunikation in einem Token-Ring-Netzwerk. Die sendebereite Station hängt ihr Datenpaket an das zu einem Besetzttoken umgewandelte Freitoken, sobald das Freitoken an dieser Station vorbeikommt. Das Token-Daten-Paar läuft durch den Ring und wird vom Empfänger kopiert. Daraufhin läuft es zurück zum Sender. Dieser löscht es und generiert ein neues Token, das wieder im Ring zirkuliert. Auf diese Weise kann immer nur eine Station im Ring senden. Die kreisförmigen Pfeile in den Abbildungen zeigen die Kommunikationsrichtung im Token-Ring an. Weitere Erläuterungen siehe Text S. 159ff.

dem nicht so sein, muß der Ring von Daten »gereinigt« und ein neues Freitoken generiert werden. Dies entspricht einem Reset des Netzwerkes.

Kreist ein Besetzttoken endlos im Ring, kann keine Station Daten versenden. Dieser Fehler kann immer dann auftreten, wenn die letzte Sendestation im Ring das Freitoken falsch generiert hat. Daher muß man in einem *Token* an einer anderen Stelle des Rings ein zusätzliches Bit setzen, das man beim nochmaligen Durchgang des *Tokens* an dieser Ringposition auf das zuvor gesetzte Bit überprüft. Ist es vorhanden, so muß der Ring ebenfalls in den Anfangszustand versetzt und ein neues »Freitoken« erstellt werden.

Ferner muß auf genau die gleiche Weise verfahren werden, wenn gerade zwei Stationen Daten versenden. In einem *Token*-Ring darf immer nur eine Station senden. Folglich muß jede Station überprüfen, ob die Nachricht, die gerade ankommt, auch die gleiche ist, die zuvor abgeschickt wurde. Ist dem nicht so, senden zwei Statio-

nen gleichzeitig. Daher muß die überprüfende Station vorbereitete Sendevorgänge sofort abbrechen, darf jedoch kein neues *Token* erzeugen. Dies führt zur Generierung eines Fehlers, nämlich daß *kein Token* existiert. Erst dann kann der Fehler durch einen Reset des Rings beseitigt werden.

Den normalen Ablauf in einem Ringsystem stellt *Abb. 7* noch einmal zusammenfassend dar.

Die richtige Topologie
Für den Endanwender kommt sowohl das Ring- als auch das Bussystem als Topologie zum Einsatz in Betracht. Das *Bussystem* weist jedoch die meisten Vorzüge auf, was an der einfacheren Realisierung liegt. Ein *Ethernet*-System läßt sich relativ schnell aufbauen, und weitere Maschinen lassen sich einfach in das Netz integrieren. Ein Ring hingegen muß aufgetrennt werden, um neue Stationen in das Netz zu einzufügen. Auch die räumlichen Begebenheiten spielen eine Rolle, wenn es darum geht, die richtige Topologie auszuwählen. Wir werden im 2. Teil des Workshops unterschiedliche Netze aufbauen und deren Vor- und Nachteile sowie deren Eignung für unterschiedliche Einsatzzwecke und Netzwerkgrößen genauer betrachten.

Das ISO-OSI-Referenzmodell
Wir haben bislang den physikalischen Aufbau eines *LANs* betrachtet und uns dabei mit zwei der bekanntetsten und der für unsere Bedürfnisse interessanten Topologien beschäftigt. Dabei wurden zwei Protokolle untersucht, welche auf den beiden Netzwerkformen basieren. Derartige Protokolle sind, je nachdem welche der beiden Topologien Sie einsetzen, immer vorhanden und aktiv und stellen damit eine der unteren Protokollebenen eines Netzwerkes dar. Andere Protokolle, wie *TCP/IP*, befinden sich erst auf einer höheren Ebene. Natürlich richten sich die Einsatzbereiche der verschiedenen Protokolle nach einem bestimmten Schema. Dieses Schema ist das *ISO-OSI-Referenzmodell*, das in diesem Abschnitt erklärt wird und den theoretischen Grundlagenteil unseres Workshops abschließt. Daran folgend werden wir uns intensiver im *TCP/IP* beschäftigen und dieses Protokoll ebenfalls in das *ISO-OSI-Referenzmodell* einordnen.

Die *ISO* (*International Standards Organization*) und ihre Bedeutung ist uns bereits im Kapitel um die CD begegnet. Hier betrachten wir das *OSI-Modell* (*Open Systems Interconnection Modell*), das ebenfalls von einem Gremium der *ISO* entworfen wurde und festlegt, wie sich ein Produkt zur Kommunikation mehrerer Rechner untereinander aufbaut, aus welchen Schichten es sich zusammensetzt und welche Protokolle den einzelnen Bestandteilen zugeordnet werden. Bevor wir uns das *ISO-OSI-Modell* genauer ansehen, sei noch darauf hingewiesen, daß man in der Praxis eine 1:1-Übertragung dieses Modells in die Realität nie auffinden wird. Es ist ein Referenzmodell, das eher weisenden als obligatorischen Charakter trägt. Grundsätzlich existieren auch Kommunikationsstandards, die sich nicht an das *OSI-Modell* halten. Wir werden später *TCP/IP* in dieses Modell einordnen und zeigen, auf welcher Ebene dieses Konzeptes die bereits besprochenen Topologien mit ihren Protokollen zu finden sind.

Die sieben Schichten des OSI-Modells
Das *ISO-OSI-Modell* ist in *Abb. 8* graphisch dargestellt. Wie Sie sehen, setzt es sich aus sieben Schichten zusammen, welche gemeinsam eine Kommunikation zwischen zwei Computersystemen idealisiert darstellen.

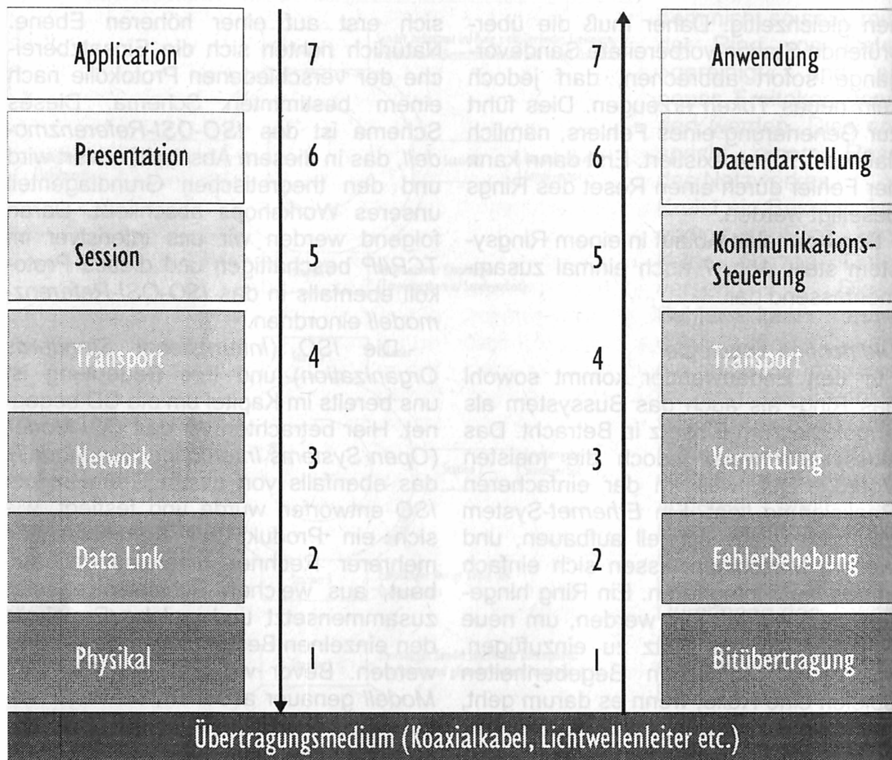


Abb. 8: Das ISO-OSI-Referenzmodell und seine 7 Schichten. Links sind die im Modell genannten Bezeichnungen angegeben, rechts deren Bedeutungen. Die einzelnen Schichten sind vom Übertragungsmedium bis zur Applikation durchnumeriert. Der Informationsfluß beginnt beim Benutzer und seiner Anwendung auf Ebene 7 und durchläuft nacheinander alle Schichten bis zum Übertragungsmedium. Jede der sieben Säulen ist als eine Station im Netzwerk zu sehen, wobei die linke Säule die sendende und die rechte die empfangende Station im Netz darstellt. Das Signal durchläuft nach der Übertragung vom Sender zum Empfänger wieder alle sieben Schichten, nur in umgekehrter Reihenfolge. Damit wird ein korrekter Informationsfluß von einem System zu einem anderen verwirklicht.

Das Modell mag anfangs etwas verwirrend wirken, ist aber sehr einfach zu verstehen, wenn man die einzelnen Schichten bestimmten Netzwerkkomponenten zuordnet.

Die beiden untersten Schichten 1 und 2 sind meistens bereits durch die zum Einsatz kommende Topologie und die darauf basierende Hardware definiert und bieten auf diesen Ebenen die bereits besprochenen Datenfluß- und Steuerungsprotokolle, die durch die Netzwerkadapter zur Verfügung gestellt werden. Auf der Ebene 3 findet man

bereits hardwareunabhängige Protokolle, welche sich u.a. um die Adressierung der einzelnen Netzstationen kümmern. Die Ebenen 4 und 5 dienen dem eigentlichen Datentransport und der Kommunikationssteuerung, während die Schichten 6 und 7 in den Anwendungsbereich fallen. Auf diesen Ebenen laufen die eigentlichen Netzwerkanwendungen mit ihren spezifischen Datenformaten. Man kann daher die sieben Schichten in drei Gruppen einteilen:

1. Schichten zum Zugriff auf das Übertragungsmedium, zur Implementierung topologieabhängiger Protokolle und hardwareunabhängiger Adressierungsschemata (Schichten 1 bis 3).
2. Schichten zur Kommunikationssteuerung und zum Datenaustausch (Schichten 4 und 5).
3. Schichten zur Ausführung von Netzwerkanwendungen und zur Verwaltung und Verarbeitung anwendungsspezifischer Datenformate.

Zur näheren Untersuchung des Referenzmodells verfolgen wir nun den Weg einer Nachricht von einer sendenden Station zu einer empfangenden Station in einem Netz.

Nehmen wir dazu an, ein Anwender verschickt eine Mail an einen anderen Benutzer. Dazu dient eine Anwendung, welche das Abfassen der Nachricht ermöglicht. Wir befinden uns also in der Schicht *Anwendung (Application)*. Nach dem Editiervorgang wird die Nachricht in einer Datei gespeichert und mit für die Datenübertragung geeigneten Zusatzinformationen versehen, etwa einem entsprechenden Header. Dies erfolgt in der Schicht *Datendarstellung (Presentation)* ebenfalls durch die Anwendung selbst oder einen anderen Prozeß durch Konvertierung des Textes in ein anderes Format, der Überwachung der Dateieingabe etc. Anschließend ist die Nachricht zur Übertragung bereit. Nun übernimmt ein Prozeß der Schicht *Kommunikationssteuerung (Session)* die Kontrolle des Kommunikationsablaufes. Hier werden bestimmte Parameter und die Flußsteuerung des Datenverkehrs initialsiert und später überwacht. Daraufhin kann die Datei übertragen werden und zwar durch die in der Schicht *Transport* definierten Protokolle. In dieser Schicht wird die gesamte Übertragung kontrolliert und durchgeführt. So wird die Datei z.B. gemäß eines bestimmten Protokolls in einzelne Datenpakete zerlegt und darauf geachtet, daß alle Pakete korrekt und in der richtigen Reihenfolge übertragen werden. Die Auflösung der Empfängeradresse und die letzten Schritte zur endgültigen Übertragung, sowie den Ablauf der Übertragung selbst übernimmt ein Protokoll (wie z.B. IP) in der Schicht *Vermittlung (Network)*. Erst jetzt wird Datenpaket für Datenpaket den Komponenten der Schichten *Fehlerbehebung (Data Link)* und *Bitübertragung (Physical)* übergeben, meistens gleich dem Treiber des Netzwerkadapters. Der Adapter selbst sichert und kontrolliert mit den bereits bekannten topologiebasierenden Protokollen den Zugriff auf das Übertragungsmedium (das ist z.B. ein Koaxialkabel), um alle Datenpakete in einem korrekten Bitstrom zu übertragen. Einmal in das Medium eingespeist, befindet sich die Nachricht im Netz und wird von der Empfangsstation paketweise aufgefangen. Hier verläuft der gesamte Prozeß während des Empfangs noch einmal -nur in umgekehrter Reihenfolge- bis aus dem erzeugten Bitstrom wieder die zuvor abgefaßte Nachricht geworden ist.

TCP/IP im Detail
Nach dem Grundlagenteil dieses Artikels soll *TCP/IP* als das Netzwerkprotokoll, das wir vorrangig verwenden werden, in den Mittelpunkt unserer Aufmerksamkeit rücken. Wir benötigen die hier wiedergegebenen Grundkenntnisse für die Praxis im zweiten und dritten Teil dieses Workshops, v.a. was die Netzwerkplanung und die Adressierung der einzelnen Arbeitsstationen mit *IP-Adressen* angeht.

TCP/IP - eine Protokollfamilie
Die Abkürzung *TCP/IP* steht für *Transmission Control Protocol/ Internet Protocol* und stellt kein Protokoll im eigentlichen Sinne, sondern ein Sammlung unterschiedlicher, miteinander verwandter Protokolle dar. Ein direkter Vergleich mit anderen Architekturen, die zur Rechnerkommunikation eingesetzt werden, ist daher nicht ohne weiteres möglich. Das liegt zum einen daran, daß TCP/IP nicht aus der Hand eines Herstellers stammt und ein einheitliches Produkt darstellt. Vielmehr wurde TCP/IP -im Anstoß an die Bedürfnisse des amerikanischen Verteidigungsministeriums- von mehereren Universitäten entwickelt und befindet sich auch jetzt unter einer beständigen Weiterentwicklung. Es ist daher notwendig, die einzelnen *TCP/IP*-Komponenten aus der Protokollfamilie zu isolieren und deren Aufbau zu untersuchen, um ein Verständnis für das Gesamtkonzept zu erhalten.

Grundlegender Aufbau
Hinsichtlich der Struktur von *TCP/IP* stellen auf diesem Protokoll basierende Netze *Peer-to-Peer-Netzwerke* dar. Das bedeutet, daß zwischen Client und Server nicht unterschieden wird. Man nennt daher die einzelnen Stationen in einem *TCP/IP*-Netz *Hosts*. Das bedeutet, daß bei der Installation eines Netzes noch keine spezifischen Serverdienste existieren. Erst durch die spätere Konfiguration des *TCP/IP*-Paketes wird bestimmt, welcher Rechner z.B. Druck- oder Dateidienste anbietet.

Trotz des *Peer-to-Peer*-Aufbaus existiert auch unter *TCP/IP* eine Client/ Server-Struktur. Unter Server versteht man hier jedoch keine Maschine, sondern ein Programm, das bestimmte Dienstleistungen anbietet, etwa das Bereitsstellen von Dateien. Diese Programme laufen in sogenannten *Dämon-Prozessen*. Das sind Prozesse, die im Hintergrund laufen, entsprechend der *TCP/IP*-Konfiguration auf einem *Host* automatisch gestartet werden und kein Interface zur Interaktion mit dem Benutzer besitzen. Für jede Serverfunktion ist ein spezifischer *Dämon* erforderlich, etwa *ftpd.exe*, der FTP-Dämon zur Bereitstellung eines Dateiservers. Ferner existiert zu jedem Server ein Client in Form eines entsprechenden *TCP/IP*-Clientprogramms, welches als normales Programm läuft, entsprechende Protokoll verwendet und mit dem *Dämon* auf einem anderen Host kommuniziert. Client und Server bezeichnen unter *TCP/IP* also Programme. Wir werden im folgenden einen genaueren Blick auf diese Programme und die Protokolle, die ihnen zugrundeliegen, werfen.

Einordnung in das OSI-Modell
Wie gesagt besteht *TCP/IP* aus einer ganzen Gruppe unterschiedlicher Protokolle, die alle aufeinander aufbauen. Auch hier existieren also verschiedene Schichten, die man in das *OSI-Modell* einordnen kann (*s. Abb. 9*).

Wie man sieht, umfasst *TCP/IP* mit einigen Komponenten gleich mehrere Schichten des Modells, einige auch gar nicht. So werden die Schichten 1 und 2 (*Bitübertragung* und *Fehlererkennung*) bereits durch die Hardware zur Verfü-

Application	7	
Presentation	6	Telnet, FTP, HTTP, NFS, DNS etc.
Session	5	
Transport	4	TCP, UDP
Network	3	IP
Data Link	2	Ethernet, Token Ring, Token Bus etc.
Physical	1	
Übertragungsmedium (Koaxialkabel, Lichtwellenleiter etc.)		

Abb. 9: Die Einordnung der TCP/IP-Protokollfamilie in das ISO-OSI-Referenzmodell. Die beiden unteren Schichten werden nicht von TCP/IP sonden von der Netzwerkhardware, also den Adapterkarten und den dazu nötigen Treibernimplementiert. Die Netzwerkschicht wird von IP (Internet Protocol) belegt. Darauf bauen die für den paketweisen Datentransport nötigen Protokolle TCP und UDP auf. Die verbleibenden Schichten 5-7 werden von den einzelnen TCP/IP-Protokollen und ihren Anwendungen, wie Telnet, FTP etc., belegt.

gung gestellt, also durch die für das Netzwerk gewählte *Topologie*. Entscheidet man sich etwa zum Aufbau eines Token Rings, bieten die Netzwerkadapter bereits eine feste Verknüpfung zwischen den beiden ersten Schichten des Modells. Unter OS/2 stellen die *MPTS (**M**ulti-**P**rotocol **T**ransport **S**ervices)* die Interfaces zwischen Adapter, Protokollen und Programmierschnittstellen bereit. Die *MPTS* beinhalten den *LAPS (**L**AN **A**dapter and **P**rotocol **S**ervice),* der Netzwerkkartentreiber und die dazugehörigen Protokolle bietet. Wir werden die *MPTS* im nächsten Teil dieses Workshops genauer untersuchen.

Auf dieser durch das Betriebssystem definierten Schicht setzt das *IP (Internet Protocol)*, die wichtigste Komnponente von *TCP/IP* auf. *IP* bildet eine Verbindung zwischen den hardwareabhängigen Protokollen und der Transportschicht, auf der sich *TCP* und *UDP* befinden. *IP* bildet die grundlegende

Funktion zum Transfer von Datenpaketen von einem *Host* zum anderen und ist auch für das sogenannte *Routing* (also das Auffinden des Weges vom sendenden zum empfangenden *Host*) verantwortlich. *IP* ist ein unzuverlässiges Protokoll, d.h. es verfügt über keinerlei Sicherheitsmechanismen, die über eine korrekte Übertragung einzelner Datenpakete im Netz wachen. Diese Aufgabe muß von einer höheren Protokollschicht erfüllt werden. Um Datenpakete zu verschicken, setzt *IP* die sogenannten *IP*-Adressen der beiden Kommunikationspartner voraus. Diese Adresse hat eine feste Länge und ist ein 32-bit Wert. Um diese Adresse in dezimaler Schreibweise besser entziffern zu können, teilt man sie in vier Oktetts und trennt diese mit drei Punkten. Eine gültige *IP*-Adresse wäre z.B. *194.120.121.5* (wobei diese Adresse rein fiktiv ist). Diese Adressen sind Ihnen sicher schon bei der Konfiguration Ihres Internetzugangs aufgefallen. Eine solche *IP*-Adresse ist keine Hardwareadresse. Damit *IP* eine entsprechende Umwandlung vornehmen kann, bedient es sich des *ARP (Address Resolution Protocol)*. Sollte die Hardwareadresse eines *Hosts* noch nicht bekannt sein, wird ein sogenannter *ARP-Broadcast* in das Netz geschickt. Nur der *Host*, dessen *IP*-Adresse im *Broadcast* angegeben ist, meldet sich, so daß *IP* nur an diesen *Host* das zu verschickende Datenpaket sendet. Ein solches Datenpaket besteht aus einem Dateiteil und einem Header, der die *IP*-Adressen des sendenden und empfangenden *Hosts* enthält. Dieser Header wird durch eine Prüfsumme gesichert, allerdings nimmt *IP* sonst keinerlei Kontrollen des Datenflusses vor.

Auf der Transportschicht finden wir die beiden Übertragungsprotokolle *UDP* und *TCP*. *UDP* bietet wie *IP* einen ungesicherten Versand[1] von Datenpaketen, erlaubt jedoch, im übrigen wie *TCP*, nicht nur den Versand an bestimmte *Hosts*, sondern zusätzlich auch an bestimmte Prozesse, welche auf diesen *Hosts* laufen. Zur Bestimmung des spezifischen Empfängers existiert eine sogenannte Portnummer, die beim Versand des Datenpaketes mit angegeben wird. Diese Zuweisungen eines Datenpaketes mit einem Port nennt man *Well Known Port Assignments*. Einige dieser Ports kennen Sie ebenfalls, etwa den TCP-Port 110 (POP3), den Sie in Ihrem POP-Mailprogramm angeben. Da die Portzuweisungen ab und an in der Praxis auftauchen, finden Sie entsprechende Tabellen in *Teil 2* dieses Workshops.

TCP (Transmission Control Protocol) nun ist das wichtigste und bekannteste Transportprotokoll. Zusammen mit *IP* hat es der Protokollfamilie auch ihren Namen gegeben. Es implementiert eine sichere Paketübermittlung von einem *Host* zum anderen, indem es einen Datenstrom, z.B. eine Datei, zum Versand in einzelne Pakete (sogenannte Datagramme) zerlegt, mittels *IP* zur Zielstation überträgt und dort wieder zusammensetzt. Dabei führt *TCP* unterschiedliche Überprüfungen durch, um die Richtigkeit des übertragenen Datenpaketes zu gewährleisten. Sollten Fehler auftreten, wird das fehlerhafte Datagramm nochmals übertragen. Eine nochmalige Übertragung findet auch dann statt, wenn ein Datenpaket, das zur Rekonstruktion des ursprünglichen Datenstroms nötig ist, verlorengeht. Außerdem bringt *TCP* die einzelnen

1 Das ist nicht ganz korrekt: *UDP* fertigt vom Datenteil eines Paketes eine Prüfsumme an. Allerdings werden weitere Überprüfungen nicht vorgenommen, was für eine gesicherte Datenübertragung noch nicht ausreicht. Diese wird durch *TCP* realisiert.

Datenpakete nach der Übertragung wieder in die richtige Reihenfolge, da *IP* nur für die Übermittlung zuständig ist und auf die korrekte Reihenfolge der einzelnen Pakete nicht achtet.

Oberhalb von *TCP* befinden sich die einzelnen Programme, etwa *FTP*, *Gopher*, *Telnet* usw. Diese Programme basieren auf unterschiedlichen Protokollen, die ihre Grundlage auf *TCP* finden. Eine Zusammenstellung der wichtigsten *TCP/IP*-Programme finden Sie im Praxisteil dieses Workshops, da wir dort auch für eine Konfiguration der entsprechenden Dämon-Prozesse sorgen.

Übrigens sind alle Protokolle in den *RFCs (Request For Comments)* aufgelistet. Möchte man für ein bestimmtes Protokoll eine neue Anwendung entwikkeln, benötigt man den entsprechenden *RFC* des Protokolls, den man über das Internet beim *NIC (Network Information Center)* via Download erhält.

Datenpakete im TCP/IP-Netz

Wir wollen nun ähnlich wie bei der Besprechung des *OSI-Refernzmodells* den Ablauf einer Dateiübertragung in einem *TCP/IP*-Netz beschreiben. Den Weg eines Datenpaketes nennt man *Route* und das Aufspüren dieses Weges vom Sender zum Empfänger *Routing*. Daher werden wir uns im folgenden Abschnitt mit dem *Routing* befassen.

Nehmen wir an, eine Datei soll in einem *LAN* von einem Host A auf einen Host B mittels *FTP* übertragen werden. Dazu wählt der Benutzer des Hosts A die entsprechende Datei aus und startet die Übertragung. Daraufhin wird die Datei von *TCP* in einzelne Datenpakete zerlegt, die -mit einem Adressheader versehen- mit *IP* verschickt werden. Dazu wird von *IP* mit *ARP* die Hardwareadresse des Zieladapters bestimmt und das Datenpaket mittels der hardwareorientierten Protokolle, auf denen *IP* aufsetzt, in das Netzwerk direkt an den entsprechenden Host geschickt. Die *Route* ist dabei klar, weshalb man auch von einem *direkten Routing* spricht. Auf diese Weise wird kontinuierlich Datenpaket für Datenpaket übertragen, von Host B mit *IP* empfangen und mit *TCP* wieder zum Originaldatenstrom zusammengesetzt.

Weitaus interessanter ist jedoch der Fall, wenn sich Host A in einem anderen Netz als Host B befindet. Nun kann das *Routing* nicht mehr direkt sein, sondern muß über mehrere Hosts verlaufen, welche das Datenpaket nur weiterreichen. Man spricht dabei von *indirektem Routing*. Damit dies möglich wird, müssen alle Datenpakete als selbständige Einheit behandelt werden, wie es unter *IP* ja auch geschieht. Daraus resultiert, daß Datenpakete auch dann den Zielhost erreichen, wenn Netzwerkbereiche zwischen Host A und Host B liegender Netzwerke ausgefallen sind. Für jedes Datenpaket kann also ein anderes *Routing* durchgeführt werden. Würde unsere Datei aus sieben Paketen bestehen und nur sechs davon erreichen den Zielhost über eine normalerweise sichere *Route* von Host zu Host und Netzwerk zu Netzwerk, fordert *TCP* das fehlende Paket noch einmal an, und es kann eine andere *Route* gefunden werden, die eine korrekte Übermittlung des Datenpaketes gewährleistet. Es können für unterschiedliche Pakete auch gleichzeitig mehrere *Routen* existieren, die unterschiedlich schnell sind. Daraus folgt, daß sich die einzelnen Datenpakete nicht in der korrekten Reihenfolge auf unserem Beispielhost B einfinden müssen. In einem solchen Falle sorgt *TCP* dafür, daß die Pakete wieder in der richtigen Reihenfolge zum ursprünglichen Datenstrom zusammengesetzt werden.

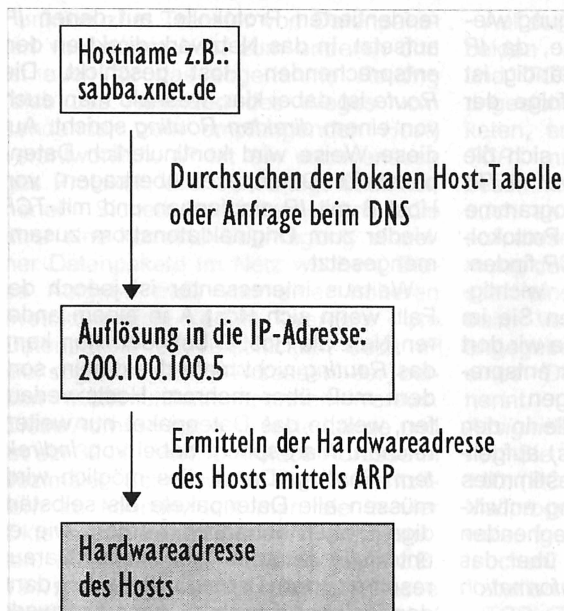


Abb. 10: Schematische Darstellung des Adressübersetzungsmechanismus des TCP/IPProtokollstacks.

Die Routen, welche die Datenpakete genommen haben, bleiben der Transportschicht *TCP* allerdings unbekannt.

Bevor wir uns mit dem hier vorgestellten Ablauf des Routings genauer beschäftigen, wird es notwendig zu verstehen, wie die *IP*-Adressierung funktioniert.

Adressierung der Netzwerkstationen
Wie wird ein Host in einem *TCP/IP*-Netz adressiert? Diese Frage führt uns zunächst zum Aufbau einer *IP*-Adresse.

Egal, welches Protokoll man in einem Netzwerk verwendet, man benutzt zwei Ebenen, um eine Netzwerkstation anzusprechen: Eine *logische*, bei der man eindeutige und leicht zu merkende Namen vergibt; und eine *physikalische*, auf der jeder Rechner eine netzwerkweit einmalige, numerische Kennung erhält.

Unter *TCP/IP* ist der *logische Name* einer Datenstation im Netz ein sogenannter *Hostname*. Der Name, den man hier vergibt, unterliegt -bis auf einige Ausnahmen- kaum Einschränkungen. Einen Host *xnet04* zu nennen ist ebenso möglich wie *xara* oder *cerberus*. Wir werden auf das Vergeben von Hostnamen auf Anwenderebene im Praxisteil dieses Workshops zurückkommen und belassen es an dieser Stelle mit den oben genannten Beispielen.

Ein von Ihnen vergebener Hostname ist für *IP* ein Synonym für eine IP-Adresse. *IP*-Adressen wurden bereits erwähnt und als 32-Bit Werte definiert, die man in dezimaler Schreibweise in vier durch drei Punkte getrennte Zahlengruppen darstellt. Während sich ein Hostname auch mit viel Phantasie zusammenstellen läßt, folgt die Vergabe von *IP*-Adressen einem festgelegten Schema, dem wir uns gleich genau zuwenden. Diese *IP*-Adresse muß netzwerkweit eindeutig sein (ebenso wir der Hostname). Sollte ein *LAN* direkt an das Internet angeschlossen sein, müssen die an die einzelnen Hosts dieses Netzes vergebenen *IP*-Adressen sogar weltweit eindeutig sein. Dies führt zu den verschiedenen *IP*-Adreßklassen, denen wir uns später zuwenden. Jeder *Hostname* ist mit einer *IP*-Adresse verknüpft, so daß die Zuordnung der einzelnen *Hosts* für *IP* möglich wird.

Wie man sich weiter denken kann, benutzt der *TCP/IP*-Protokollstack ein

dreistufiges Adreßübersetzungsschema, um einen vom Anwender vergebenen *Hostnamen* in eine *IP*-Adresse und diese in die Hardwareadresse der Arbeitsstation zu übersetzen. *IP* benutzt für diesen letzten Übersetzungsschritt, wie bereits wieter oben ausgeführt, *ARP*. Wie dieser dreistufige Mechanismus zur Adreßübersetzung aussieht, zeigt *Abb. 10*.

Aufbau von IP-Adressen
Von besonderem Interesse ist nun zunächst das Format von *IP*-Adressen. Wie wir bereits erwähnten, besteht eine *IP*-Adresse aus 32 Bit, also 4 Byte. Damit kann jedes Byte Werte von 0 bis 255 annehmen. In dezimaler Schreibweise trennt man jedes Byte der Adresse durch einen Punkt. Eine mögliche *IP*-Adresse könnte also so aussehen:

185.210.111.8

Wie werden diese vier Bytes nun interpretiert? Auf keinen Fall darf man annehmen, bei der *IP*-Adresse handelte es sich um eine fortlaufende Nummerierung aller im Netz vorhandenen *Hosts*. Vielmehr läßt sich das Zahlenquartett in zwei Gruppen zerlegen, wobei die erste Gruppe für die Nummer des Netzwerkes und die zweite für die des *Hosts* in diesem Netzwerk steht. Betrachten Sie dazu *Abb. 11*.

Welche der vier Bytes man für die Bezeichnung des Netzwerkes und welche für die der *Hosts* verwendet, ist abhängig von der Adreßklasse. Je nachdem, welche Bytes man zur Netzwerk- bzw. Hostnummer gruppiert, ergeben sich nämlich drei Adreßklassen: A, B und C, welche *Abbildung 11* darstellt.

Wie man sieht, wird bei Adressen der Klasse A nur das erste Byte der Adresse zur Bezeichnung des Netzwerkes verwendet, während die anderen 3 Bytes zur Numerierung der *Hosts* in den Netzen zur Verfügung stehen. Da bei Klasse-A-Adressen das erste Bit des ersten Bytes immer auf 0 gesetzt ist, lassen sich mit diesem Byte 2^7 Netze adressieren, also 128. Der Wertebereich des ersten Adreßbytes beginnt damit bei 0 und endet bei 127. Für die Adressierung der Hosts in einem der 128 Netze stehen in der IP-Adresse nun noch 3 Byte zur Verfügung. Damit können insgesamt 2^{24}, also 16.777.216 *Hosts* pro Netz eindeutig adressiert werden. In einem Netz kann man also sehr viele Hosts miteinander verbinden. Klasse-A-Adressen werden deshalb für Internet-Rechner vergeben. Bei 128 möglichen Netzen kann man so insgsamt 2.147.483.648 einzelne Hosts adressieren.

Bei Adressen der Klasse B werden die ersten beiden Bytes der *IP*-Adresse zur Bezeichnung des Netzwerkes und die letzten beiden zur Bezeichnung der *Hosts* in diesen Netzen verwendet. Das Verhältnis zwischen Netzen und Hosts ist damit recht ausgewogen. Die ersten beiden Bit einer Klasse-B-Adressen werden immer auf 10 gesetzt, womit sich für das erste Byte ein Wertebereich von 128 bis 191 ergibt. Insgesamt stehen noch 6 Bit des ersten Bytes und 8 des zweiten zur Netzwerkadressierung zur Verfügung, womit insgesamt 2^{14}, also 16.384 Netze adressiert werden können. Da die beiden letzten Bytes der Klasse-B-Adresse vollständig zur Adressierung der *Hosts* in einem Netz verwendet werden können, ist es möglich pro Netz 2^{16}, also 65.536 Hosts zur adressieren. Insgesamt bieten Klasse-B-Adressen einen Adressenpool von 1.073.741.824 Hosts.

Die am meisten verwendeten Adressen sind Klasse-C-Adressen, die im Internet übrigens nicht verwendet wer-

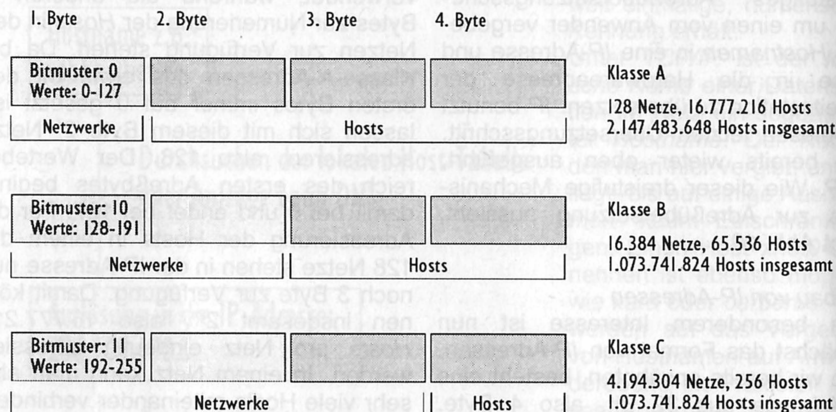


Abb. 11: Die drei Adreßklassen des TCP/IP-Protokollstacks. Unter Bitmuster versteht man bei der Klasse A das erste, bei den anderen Klassen die ersten beiden Bits des ersten Bytes der Adresse.

den. Man wird daher diesen Adreßtyp in seinem eigenen *LAN* am wahrscheinlichsten einsetzen. Bei Klasse-C-Adressen werden die ersten drei Bytes zur Adressierung des Netzwerkes benutzt und das vierte zur Adressierung der *Hosts*. Die ersten beiden Bits des ersten Bytes sind immer auf 11 gesetzt, womit sich der Wertebereich 192 bis 255 für das erste Byte ergibt. Insgesamt stehen also noch 22 Bits zur Adressierung der Netzwerke zur Verfügung. Damit läßt sich eine große Anzahl an Netzwerken beziffern, nämlich 2^{22}, also 4.194.304. Dafür stehen nur noch 8 Bit zur Hostadressierung pro Netzwerk bereit. Ein *TCP/IP*-Netz, das Klasse-C-Adressen benutzt, kann daher aus höchstens 256 *Hosts* bestehen. Dadurch bleiben die Netze aber überschaubar. Da man mehrere Klasse-C-Netze -wie auch die Netze anderer Klassen- miteinander koppeln kann, lassen sich mit dieser Adreßklasse genauso viele Hosts adressieren wie bei der Verwendung von Klasse-B-Adressen, also 1.073.741.824.

Um diesen Abschnitt abzuschließen, sei noch auf zwei Besonderheiten hingewiesen, nämlich auf *Netz- und Broadcastadressen*. Eine *Netzadresse* wird durch eine *IP*-Adresse repräsentiert, bei der die Bereiche, welche die *Hostadresse* kennzeichnen, auf 0 gesetzt sind, etwa 132.223.0.0 bei einer Klasse-B-Adresse. Daher kann man eine derartige Adresse nicht für die Adressierung eines *Hosts* verwenden. Bei einer *Broadcastadresse* werden die den Host bezeichnenden Adreßteile auf 255 gesetzt. Damit werden alle Bits dieses Adressteiles auf 1 gesetzt. Eine Klasse-A-Broadcastadresse wäre z.B. 22.255.255.255. Mit einer derartigen Adresse lassen sich alle Rechner im Netzwerk auf einmal ansprechen. Auch eine solche Adresse darf nicht zur Adressierung eines *Hosts* Verwendung finden.

Routing durch mehrere Netze

Wenn wir uns nun noch einmal dem Routing zuwenden, können wir mit Hilfe des Wissens um die *IP*-Adressierung

rasch verstehen, wie die Wegfindeung zu einem Zielhost von einem Netz zum anderen im Detail funktioniert.

Ein *direktes Routing* findet immer dann statt, wenn sich der *Zielhost* im gleichen Netzwerk befindet, wenn also die Netzwerknummer der *IP*-Adresse beider Kommunikationspartner dieselbe ist.

Stellt sich bei der Übersetzung der *IP*-Adresse in eine Hardwareadresse heraus, daß sich der *Host* nicht im gleichen Netz wie die sendende Station befindet, muß das zu verschickende Datenpaket *indirekt geroutet* werden. Dazu muß das *TCP/IP*-Netz über einen **Default-Router** verfügen. Das ist ein *Host*, an den alle Datenpakete geschickt werden, wenn ein sendender *Host* keine andere *Route* zum Ziel kennt. Ferner muß ein *Router* ein *Host* mit mindestens zwei Netzwerkinterfaces (also Netzwerkkarten) sein, der mit zwei verschiedenen Netzwerkadressen konfiguriert ist. Praktisch heißt das, daß ein Router zwei *IP*-Adressen besitzt, die sich im Netzwerkteil der Adresse unterscheiden. Damit bildet ein *Router* ein Interface zwischen zwei Netzwerken. Interessant ist dabei, daß man nicht erkennen kann, was sich hinter dem *Router* befindet. Diesen Umstand kann man sich auch zunutze machen, wenn man ein Netzwerk hinter einem anderen verbergen möchte: Man kann Nachrichten von einem Netz über den *Router* ins andere schicken, ohne einen Zugriff auf dieses Netz zu haben. Wir beschäftigen uns mit dieser Art des *Routings* im dritten Teil unseres Workshops.

Erhält der *Router* nun ein Datenpaket, das an einen *Host* adressiert ist, der sich im Netzwerk des *Routers* befindet, kann er das Paket direkt an den entsprechenden *Host* senden. Sollte sich dieser *Host* aber nicht im Netz befinden, schickt der *Router* die Nachricht an den *Default-Router* des eigenen Netzes. Dieser hat genau die gleiche Funktion und arbeitet auf dieselbe Weise wie der zuvor besprochene *Default-Router*. Dieser Vorgang wiederholt sich so lange, bis das Datenpaket den *Zielhost* erreicht hat.

Die auf diese Weise gefundene *Route* wird übrigens nicht vergessen, sondern für spätere Datenübertragungen gesichert. Da *Router* mit dem Protocol *RIP (Routing Information Protocol)* Informationen mit anderen Routern austauschen können, wird die sogenannte *Routing-Tabelle* des *Hosts* immer aktuell gehalten. Sollte ein *Router* eine neue *Route* durch ein Netz finden, teilt er diesen Umstand so schnell wie möglich allen anderen bekannten *Routern* im Netz mit. Diese aktualisieren daraufhin ihre *Routing-Tabellen*, um die neue *Route* zu verwenden. Diese Methode bezeichnet man als *dynamisches Routing*, weil die *Router* ihre *Routing*-Tabellen selbst erweitern und so ihre *Routen* dem Betriebsverhalten (wie Erreichbarkeit und Auslastung) der einzelnen Netze und *Hosts*, über die Datenpakete weitergeleitet werden, anpassen können. Auch der Konfiguration von *Routern* wenden wir uns in den Praxisteilen dieses Workshops genauer zu.

Hostnamen in IP-Adressen umsetzen

Um den Bereich der Adreßübersetzung abzuschließen, fehlt nun noch die Darstellung des Verfahrens, mit dem ein *Hostname*, den Sie beim Einrichten Ihres Netzwerkes vergeben, in eine *IP*-Adresse übersetzt wird, da *IP* diese Adresse benötigt, um mittels *ARP* die Hardwareadresse des entsprechenden *Hosts* zu ermitteln. Dazu gibt es zwei Möglichkeiten: Entweder die Verwendung einer sogenannten *Hosttabelle* oder den Einsatz eines *DNS (Domain Name Server)*.

Umsetzung mit Host-Tabellen
Wir wissen, daß ein *TCP/IP*-Netz vom Aufbau her ein *Peer-to-Peer-Netz* ist. Folgerichtig muß jeder *Host* die Namen der anderen *Hosts* kennen, mit denen er kommunizieren möchte. Da es ferner notwendig ist, die vom Administrator vergebenen *Hostnamen* *IP*-Adressen zuzuordnen, befindet sich auf jedem *Host* eine *Host-Tabelle*, welche diese Zuordnungen aller *Hosts* im Netz beinhaltet. Diese Datei ist eine reine Textdatei und befindet sich im Verzeichnis *\tcpip\etc* unter dem Namen *hosts*.

In dieser sich auf jedem *Host* befindenden Datei findet man Zuordnungen nach folgendem Muster:

```
# Adresse        Name    Alias   Kommentar
193.120.120.10 xnet01  sabba   # Testhost
193.120.120.11 xnet02  xara    # Dateien
193.120.120.13 xnet03  belana  # Ersatz
```

Die erste Spalte der Tabelle bezeichnet die bei der Netzwerkplanung vergebene *IP*-Adresse. Die zweite Spalte definiert einen Namen, unter dem der *Host* im Netzwerk angesprochen werden kann. Die dritte Spalte erlaubt die zusätzliche Definition eines *Alias*, also eines Zweinamens für den *Host*. In der vierten Spalte kann ein Kommentar angegeben werden. Wenn man etwa den *Host xnet03* ansprechen möchte, könnte man auf der Kommandozeile folgenden Befehl eingeben:

```
ping xnet03
```

oder, was man sich besser merken kann:

```
ping belana
```

TCP/IP greift bei Eingabe eines derartigen Kommandos auf den sog. *Resolver* (eine Bibliotheksfunktion) zurück, der in unserem Beispiel nach dem Namen *belana* in der Host-Tabelle sucht und ihn übersetzt, so daß aus dem Kommando ein:

```
ping 193.120.120.13
```

wird. Der Vorteil, eine *Hosttabelle* zu verwenden, liegt v.a. in der einfachen Pflege. Sollte man einen Rechner aus dem Netz entfernen oder einen neuen einbinden, kann man die Tabelle recht einfach editieren. Vielleicht mag es mühsam erscheinen, die gleiche Tabelle auf jedem *Host* anzulegen; doch in der Praxis macht man dies -wenn überhaupt- nur einmal, nämlich bei der Einrichtung des Netzwerkes. Später wird die Tabelle auf nur einem *Host* gepflegt und von dort mittels *FTP* auf alle anderen *Hosts* übertragen. Man kann derartige Übertragungen bereits beim Systemstart von OS/2 einrichten, um stets eine aktuelle *Host-Tabelle* zur Verfügung zu haben. Wie das funktioniert, werden wir in den folgenden Artikeln dieses Workshops sehen.

Umsetzung durch einen DNS
Eine andere Möglichkeit, die *Hostnamen* entsprechend aufzulösen, liegt in der Einrichtung eines *DNS (Domain Name Servers)*. Ein *DNS* ist ein *Host* in einem *TCP/IP*-Netzwerk, der eine *Hostdatei* pflegt und sie anderen Systemen bei Bedarf zur Verfügung stellt. Alles, was die anderen *Hosts* des Netzwerkes wissen müssen, ist lediglich *eine* Adresse, nämlich die des *DNS*, die in der Datei *resolv* im Verzeichnis *\tcpip\etc* zu finden ist. Wie man sich sicher vorstellen kann, existiert auch hierfür ein Protokoll, d.h. es gibt einen *DNS-Client* und einen *DNS-Server*. Bei unserem Beispiel mit dem kleinen Programm *ping.exe* weiter oben, würde der *Host* über den *Resolver* den in der Datei *resolv* angegebenen *DNS* zu erreichen

versuchen, ihm den Namen des Zielhosts übermitteln und dafür, sofern ein Eintrag beim *DNS* besteht, die *IP*-Adresse zurückbekommen. Sollte die Antwort des *DNS* negativ sein, läßt sich also kein Eintrag für diesen Namen *im zentralen Hostverzeichnis* finden; oder sollte der *DNS* unerreichbar sein, versucht der anfragende *Host* in seiner *lokalen Host-Tabelle* das Zielsystem ausfindig zu machen.

Ein *DNS* verringert den administrativen Aufwand, v.a. bei sehr großen Netzwerken, weil man Hostinformationen nur an einer Stelle pflegen muß. Allerdings sollte man dennoch lokale *Host-Tabellen* für den Fall zur Verfügung stellen, daß der *DNS* ausfällt. Wie man einen *DNS* konfiguriert, soll ebenfalls Bestandteil der nächsten Teile dieses Workshops sein.

Domains

Zum Schluß bleibt nur noch das Konzept der *Domains* einzuführen.

Wir haben weiter oben einige *Hostnamen* in unseren Beispielen vergeben. Damit decken wir den Teil der *IP*-Adresse leicht einprägsam ab, der die einzelnen *Hosts* adressiert. Um nun eine weitere, logische Gruppierung vornehmen zu können, muß man noch das Netzwerk, in dem sich die *Hosts* befinden, benennen. Dies geschieht mit *Domainnamen*.

Eine *Domain* ist nichts anderes als ein Rechnerverbund, also ein Netz. Man kann den Namen hier ebenso frei wählen wie die Bezeichnungen der *Hosts*, es sei denn, die *Domain* ist an das Internet angeschlossen. Hier muß ebenfalls sichergestellt sein, daß ein bestimmter *Domainname* nicht doppelt vergeben wird. Der *Domainname* unterteilt sich außerdem noch weiter und zwar in eine *Toplevel-* und eine *Sublevel-Domain*. Als Beispiel betrachten wir den bereits genannten Host *belana*, der sich im Netzwerk unseres Verlages befindet. Die vollständige Adresse wäre hier:

belana.xnet.de

Der Host *belana* befindet sich in einem Netzwerk, das *xnet* heißt. Der Name *xnet* ist die *Subleveldomain*. Alle *Hosts*, die sich im gleichen Netzwerk befinden, verfügen über die gleiche *Subleveldomain*, führen in ihrem Namen also die Bezeichnung *xnet*. Die *Topleveldomain* wird durch das Kürzel *de* gekennzeichnet. Sie kennen *Topleveldomains* aus dem Internet. Bekannt sind ebenfalls *.org*, *.net*, *.com* usw. Unsere *Topleveldomain* ist das Länderkürzel für Deutschland. Für ein lokales Netz, das nicht direkt mit dem Internet verbunden ist, müssen Sie sich jedoch nicht an diese Nomenklatur halten, was wir in den Praxisteilen zeigen werden. Hier ist nur wichtig zu beachten, daß man seinen Netzen -wie den *Hosts*- ebenfalls Namen gibt, um sie eindeutig adressieren zu können. Diese Namen werden ebenfalls in der *Host-Tabelle* oder beim *DNS* registriert. Wichtig werden die Domainnamen dann, wenn Sie mehr als nur ein Netzwerk betreiben, da es dann erforderlich wird, die Netze ebenfalls auseinanderhalten zu können. Alle Rechner der neuen *Subleveldomains* (also der neuen Netze) gehören dann, wenn wir bei unserem Beispiel bleiben, zur Topleveldomain *de*. Mit den *Domains* kann man also die etwas kryptische *IP*-Adresse vollständig und leicht einprägsam darstellen.

Damit sind wir am Ende der ersten Teils unseres Workshops angelangt. Auf der Grundlage des vermittelten Basiswissens werden wir in den nächsten Teilen mit dem praktischen Aufbau von OS/2-Netzen beginnen. ❑

In eigener Sache

Wie auch im vorherigen Band erhalten Sie in diesem Kapitel Informationen rund um unsere Projekte *OS/2 Only!* und *Xelia* sowie Hinweise zu Terminen und neuen Projekten. In dieser Ausgabe informieren wir Sie v.a. über die zukünftige Erscheinungsweise der *OS/2 Only!*, so daß wir Sie bitten, die folgenden Abschnitte aufmerksam durchzulesen.

Verzögerungen und ihre Gründe

Wie schon im Begleitschreiben zu diesem Band erwähnt, möchten wir kurz auf die Verzögerungen eingehen, von denen Sie in der letzten Zeit betroffen waren.

Mit dem Erfolg der *OS/2 Only!* kamen auf uns einige Schwierigkeiten zu, und als Leser der *OS/2 Only!* sind Sie berechtigt, einen Einblick in unsere Unternehmensstruktur nehmen zu dürfen, mit dem unsere Arbeit besser verständlich wird. Wir sind eigentlich ein belletristischer Publikumsverlag. Die Vorbereitungen für eine Novelle oder einen Roman sind nun natürlich denkbar anders als für ein fachliches Periodicum für OS/2, das gilt natürlich auch für andere Bereiche, wie etwa Vertriebswege oder die Präsentation im Internet. Das bedeuted, Umstrukturierungen und Investitionen werden nötig: Neue Hardware, Software, Mitarbeiter, eine andere Werbe- und Vertriebsstruktur usw. Die Beiträge, die Sie in der *OS/2 Only!* lesen, entstehen bis auf einige Ausnahmen in unserem Haus, so daß Sie sich vorstellen können, wieviel v.a. auch andersartige Arbeit auf uns zukam. Und aus dem anfangs nur zaghaft realisierten Projekt wurde eine weitaus größere Aufgabe. Durch diese Probleme können wir die anfangs geplante zweimonatige Erscheinungsweise nicht einhalten. Um Ihnen und uns immer wiederkehrende Verschiebungen der Erscheinungstermine zu ersparen, werden wir die Termingrenzen so lange wegfallen lassen, bis wir die anfallenden Neustrukturierungen in unserem Hause abgeschlossen haben.

Künftige Erscheinungsweise

Die neuen Maßnahmen bedeuten zunächst, daß die *OS/2 Only!* als neuer Betriebszweig *uneingeschränkt* weitergeführt wird. *Das Projekt wird also definitiv* ***nicht*** *eingestellt!* Sie brauchen somit keine Befürchtungen haben, nach dem ersten und zweiten Band sei es aus mit dem neuen Magazin. Es geht weiter, wenn zunächst auch etwas langsamer als zunächst vorgesehen. Sie werden in diesem Jahr auch noch die vorgesehenen Bände erhalten, aber nicht alle zwei Monate. Das heißt: *Die folgenden Bände der OS/2 Only! erscheinen in zwangloser Folge*. Da die *OS/2 Only!* zum größten Teil aus Artikelserien und Workshops besteht, sieht unser neues Programm vor, nun nicht mehr Ausgabe für Ausgabe vorzubereiten, wie beim *1.* und *2. Band*, sondern Beitrag für Beitrag. Auf diese Weise können wir, bis auf das Kapitel *Neuigkeiten*, die Arbeiten an mehreren Bänden parallel abschließen. Damit erscheinen die folgenden Bände der *OS/2 Only!* nach der üblichen Vorbereitungszeit in kürzeren Abständen, also relativ dicht gedrängt im Herbst/ Winter dieses Jahres, beginnend etwa ab Ende August. Genaue Erscheinungstermine können wir jetzt allerdings noch nicht bekanntgeben. Bis Jahresende dürfte es möglich sein, für den *2. Jahrgang* die geplanten Erscheinungstermine vom Februar 2000 an einhalten zu können. Wenn also im *1. Jahrgang* Verzögerungen auftreten, und feste Erscheinungstermine nicht festgesetzt werden kön-

nen, da sie für den Anfang unter realistischen Gesichtspunkten nicht einhaltbar sind, so bitten wir Sie um ein wenig Geduld: Für uns **und** das Projekt. Wir denken, es wird sich lohnen. Qualität ist ein weitaus wichtigerer Gesichtspunkt als Quantität.

Preise und Konditionen

Preise, Liefer- und Zahlungsbedingungen verändern sich selbstverständlich nicht. Auch nicht in einem langfristigen Rahmen. Abonnements gelten weiterhin, d.h. ein Abonnement umfaßt 6 Ausgaben. Unsere Abonnenten werden, wie mit dem 2. Band, automatisch bei Erscheinen der anderen Bände mit den neuen Ausgaben frei Haus beliefert. Zusätzliche Gebühren oder dergleichen fallen nicht an.

Zu den Abonnementsbestellkarten wurden wir gefragt, ob diese auch gefaxt werden können. Das ist selbstverständlich möglich: Karten und Formulare für Bestellungen und Vormerkungen aller Art können Sie uns natürlich auch zufaxen, was in vielen Fällen kostengünstiger als das Verschicken der Karte per Post ist. Bis auf die Abonnements können Sie Bestellungen oder Vormerkungen auch telefonisch oder per e-Mail aufgeben. Nennen Sie jedoch in allen Fällen immer Ihre Kundennummer, sofern Sie bereits eine haben. Das beschleunigt den Bestellvorgang.

Autoren gesucht

Sie können die OS/2 Only! selbst mitgestalten. Geben Sie Ihr Wissen um OS/2 einfach weiter, wir freuen uns über neue Autoren, zumal jeder Beitrag, den wir von Ihnen bekommen, die Vorbereitungszeit einer Ausgabe erheblich vermindert.

Wir erhalten recht oft Anfragen zu den möglichen Themenbereichen, zu denen Sie Artikel einreichen können. Dazu sei gesagt, daß wir bezüglich der Themen keine Vorgaben machen. Sie haben sicher schon bemerkt, daß die *OS/2 Only!* einen recht großen Themenbereich abdeckt. Folglich sind Sie in der Wahl des Themas nicht gebunden. Voraussetzung ist natürlich, daß der Artikel mit OS/2 zu tun hat. Übrigens: Nicht nur Artikel mit fachlichem Inhalt sind uns willkommen. Auch über kritische Kolumnen oder Essays freuen wir uns. Da uns Ihre Meinung interessiert, können Sie auch mal den ein oder anderen Leserbrief schreiben. Einige Leser haben dies schon getan, wovon Sie sich in dieser Ausgabe überzeugen können. Sind Sie an einer tiefergehenden Autorentätigkeit interessiert, finden Sie auf unserer Homepage auf den *OS/2 Only!*-Seiten alle nötigen Informationen. Für spezielle Fragen rufen Sie uns einfach an. Artikelvorschläge können Sie auch unaufgefordert via e-Mai, Fax oder per Post an uns senden. Auf Ihre Mitarbeit freuen wir uns bereits jetzt und hoffen von allen schreibinteressierten Anwenderinnen und Anwendern zu hören. Vor allem Sondererscheinungen sind von Ihrer Mitarbeit abhängig.

Die Zukunft unseres Verlagsprogramms

In Zukunft dreht sich in unserem Verlag so gut wie alles um OS/2. Wir planen mittelfristig einen verstärkten Ausbau des Zeitschriften- und Softwarebereiches. Das betrifft nicht nur Investitionen sondern auch das Angebot neuer Serviceleistungen. Mit der Inbetriebnahme des *Xelia Network* ab Mitte nächsten Jahres steht Ihnen als Leser der *OS/2 Only!* ein kostengünstiger Online-Service zu Ihrem Magazin und zu OS/2 zur Verfügung. Wir werden einige Einrichtungen unserer Homepage in diesen neuen Online-Dienst verlegen, da auf

diese Weise ein bedeutend besseres Updateverhalten bei niedrigeren Kosten möglich ist. Da das Netz bundesweit zum Ortstarif zu erreichen sein wird, entstehen Ihnen auch keine der hier und da üblichen »0190-Hotline-Kosten«. Weitere Informationen zum *Xelia Network* finden Sie auf den *Xelia*-Seiten unserer Homepage. Einen umfangreichen Artikel zum neuen Online-Dienst lesen Sie in *Bd. 4* der *OS/2 Only!* in diesem Kapitel.

Mit dem geplanten Online-Angebot möchten wir Ihnen im Zusammenspiel mit der *OS/2 Only!* ein kompetent informierendes Informationszentrum zu OS/2 geben. Im Rahmen des *Xelia-Ideenwettbewerbes*, an dem Sie bis zum *1. September 1999* übrigens noch über unsere Homepage teilnehmen können, haben wir für das neue Netz interessante Vorschläge bekommen, die wir Ihnen im nächsten Band vorstellen werden.

Ferner erweitern wir kontinuierlich den Softwarebereich. Einige unserer Programme koppeln wir aus unserem Hauptprojekt *Xelia* aus, um Sie auf anderen Wegen anzubieten. Zu diesen Einzelprodukten gehört eine *T-Online-Suite* und ein neuer *Webbrowser*, der kostenlos über unsere Homepage zu beziehen sein wird. Wir haben uns letztlich zum kostenlosen Vertrieb dieses OS/2-Browsers entschieden, da es keine Alternative zu *Netscape* gibt, möchte man mit einem OS/2-Programm arbeiten. Das ist offenbar ein Dilemma, für das es einen Ausweg geben sollte. Den neuen Browser erhalten Sie ab Sommer in einem etwa 250 KByte großen selbstentpackenden ZIP-Archiv auf den Downloadseiten der *Xelia*-Homepage.

Unsere neue Strategie verknüpft Software und Informationen eng miteinander. Wir sind uns sicher, damit einen wichtigen Beitrag für alle OS/2-Anwender zu leisten.

Die OS/2 Only! CD Nr. 1

Zur *OS/2 Only!* gehören auch zwei CDs. Beide erscheinen wie geplant, d.h. die Sommer-CD Juni/Juli und die Winter-CD Dezember/Januar. Für die erste CD finden Sie eine Bestellkarte als Beilage in diesem Band, die wir freundlich zu beachten bitten.

Die erste CD wird eine ganze Reihe unterschiedlicher Programme enthalten, wie eine aktuelle Demo des *RSJ CD-Writers*. Wir sind auch bemüht, eine *StarOffice*-Version auf den Datenträger zu übernehmen. Diesbezüglich können wir jedoch noch nichts Genaues sagen. Neben dem ersten Teil unserer Treibersammlung finden Sie auch ein umfangreiches Archiv lohnenswerter Free-und Shareware, darunter *ISDN-PM*, die in der *OS/2 Only!* vorgestellten Programme, alles für *TCP/IP* und das *Internet*, hierunter auch die Software für unsere Netzwerkworkshops, also *TCP/IP-Client- und Serverprogramme* (wie den *Apache-Webserver*) aber auch größere Projekte wie *XFree 86*, *LaTex* und *POV-Ray für OS/2* etc. Das zum Zeitpunkt der CD-Herstellung aktuelle *OS/2-Fixpack* werden Sie ebenfalls auf dem ersten *OS/2 Only!*-Silberling finden. Neben einer Schriftartsammlung rundet die erste *Xelia*-Demoversion die *Sommer-CD '99* ab. Wie die Bände der *OS/2 Only!* bereiten wir auch die CD mit großer Sorgfalt vor. Auf unserer Homepage finden Sie bereits ein vorläufiges Inhaltsverzeichnis. Bitte beachten Sie, daß sich der Inhalt der CD bis zur Herstellung noch ändern kann. Bestellungen können Sie wie üblich direkt über unsere Homepage, mit der beigelegten Bestellkarte oder formlos per Telefon, e-Mail oder Fax aufgeben.

Weitere Sondererscheinungen?

Wir wurden von Lesern gefragt, ob neben den geplanten Sondererschei-

nungen, den beiden CDs und dem Indexband, weitere außerplanmäßige Titel zur *OS/2 Only!* erscheinen werden, etwa Bücher zu Themen wie: *»Ins Internet mit OS/2«* oder zu anderen Bereichen, welche die Publikation eines Sonderbandes rechtfertigen. Grundsätzlich ist uns die Publikation derartiger Sondererscheinungen möglich, jedoch z.Zt. nicht redaktionell. Demzufolge freuen wir uns über alle Autorinnen und Autoren, die über ein Thema schreiben möchten, welches besser in einem Sonderband behandelt werden kann als in einer normalen Ausgabe der *OS/2 Only!* Wir haben dazu einige Themenvorschläge auf unserer Homepage für Sie zusammengetragen. Wer Interesse an Sonderprojekten des *OS/2 Only!*-Bereiches hat, an einem Projekt teilnehmen oder eigene Vorschläge einbringen möchte, kann uns über das auf der Homepage zur Verfügung stehende Formular eine kurze Nachricht senden.

Ausblick auf Bd. 3

Vielleicht haben Sie bemerkt, daß wir für diesen Band einige Artikel in den nächsten verschieben mußten. Themenverlagerungen können aus aktuellen Anlaß immer auftreten. Unter diesem Einwand erwarten Sie in der *OS/2 Only! Nr. 3* folgende Informationen:

Neuigkeiten

Kurzmeldungen, Vorstellung des neuen *Warp Servers for e-business*, *Java für OS/2* installieren und einsetzen.

Hardware

Theorie: Massenspeichersysteme. Vergleiche verfügbarer Massenspeichersysteme und deren Einsatz unter OS/2.

Praxis: SCSI Workshop, Teil 1. Wahl eines geeigeneten Adapters, Tips zur Installation, IDE und SCSI parallel betreiben.

Software

Backup mit PKZip 2.50: Vorstellung einer preisgünstigen und sicheren Backuplösung mit PKZip 2.50.

LaTex für OS/2-Workshop, Teil 1. Kurzvorstellung des Programmpaketes, benötigte Software, Hilfsmittel, Installation, Konfiguration, Erzielen erster Resultate.

Know how

Theorie: Die OS/2 DOS-Unterstützung. Theroetische Grundlagen zur Unterstützung von DOS- und Windowsprogrammen unter OS/2

Praxis: DOS- und Windows-Programme einsetzen. Wartungssysteme, Teil 2, darin: Erzeugen einer Boot-CD. *PM-Log-Programm* für *chkdsk*.

Programmieren

DLL-Workshop, Teil 3: Programmierung eines Multithreading-Subsystems.

Start unserer Reihe *OS/2-Programmierung*. In diesem Band : *OS/2-Programmierung mit REXX, Teil 1.*

Netzwerk

HTML-Workshop, Teil 2: Hilfsmittel zum Erstellen von Webseiten, HTML-Sprachreferenz.

OS/2-Netzwerkworkshop, Teil 2: Benötigte Hardware und Treiber, Installation unterschiedlicher Topologien (Ethernet und Token Bus), Praxisteil zu TCP/IP, wichtige Programme, Konfiguration, Einrichtung des TCP/IP-Paketes zur Inbetriebnahme des Netzes .

Multimedia

POV-Ray für OS/2, Teil 1: Benötigte Software, Hilfmittel zur Erstellung von Szenen und Animationen, Installation, Konfiguration, Erzielen erster Resultate, POV-Ray Sprachreferenz zur Erstellung photorealistischer Szenen und Animationen. ❑

OS/2 Only!

Das fortlaufende Sammelwerk nur für OS/2

Abonnement

☐ Hiermit bestelle ich die OS/2 Only! ab der nächsten verfügbaren Ausgabe im Abonnement zum Preis von DM 136,90 (EUR 75,65 Auslandspreis) inkl. 16 % MwSt., Porto und Verpackung für 12 Monate (6 Ausgaben). Außerdem erhalte ich als Abonnent der OS/2 Only! auf alle OS/2 Only! CD-ROMs einen Preisnachlaß von 10 %. Das Abonnement ist jederzeit mit einer Kündigungsfrist von 12 Wochen kündbar. Es verlängert sich nach Ablauf eines Jahres, sollte ich nicht kündigen, automatisch um ein weiteres Jahr.

☐ Ich bin Schüler/in, Student/in, Auszubildende/r oder Wehr- bzw. Zivildienstleistender und bestelle das oben beschriebene Abonnement zu einem ermäßigten Preis von DM 124,95 (EUR 69,25 Auslandspreis). Diese Karte sende ich daher zusammen mit einem entsprechenden Nachweis ein.

Die Abonnementsgebühren sind gegen Rechnung innerhalb von 14 Tagen nach Erhalt ohne Abzug zahlbar.
Ich kann diese Abonnementsvereinbarung innerhalb von 10 Tagen beim C.-E. Fischer Buchverlag, Stuttgart, schriftlich widerrufen. Zur Wahrung dieser Frist genügt die rechtzeitige Absendung meines Widerrufs (Datum des Poststempels).

☐ Bitte informieren Sie mich außerdem über Neuerscheinungen aus Ihrem OS/2-Programm und über Software bzw. Sondererscheinungen zur OS/2 Only!

☐ Ich bin daran interessiert, auch von weiteren Anbietern Informationen zum Themenbereich OS/2 zu erhalten.

X

Datum, Unterschrift

Xelia für OS/2 ab 2.1

Vormerkung

Xelia, das ist die neue Anwendungs- und Entwicklungsarchitektur nur für OS/2 und beinhaltet über 25 integrierte Anwendungen sowie eine neue graphische Benutzeroberfläche, schön, schlank, schnell und - einfach zu bedienen! Xelia bietet eine konsequent objektorientierte Arbeitsweise, besitzt Multidesktopfähigkeiten und bietet eine ideale Architektur zur Systempflege sowohl von Einzelplatzrechnern als auch von Netzwerken. Das Programmpaket macht dabei intensiven Gebrauch von OS/2-Spezifika. Im Zusammenhang mit Xelia bieten wir ab dem Jahr 2000 auch das XeliaNetwork, einen neuen OS/2-Onlinedienst für alle Xelia-Anwender mit Software, neuen Treibern und Informationen -zum kostenlosen Download-, Serverdiensten und Internetzugang. Vorgemerkte Anwender erhalten regelmäßig Informationen, 15 % Rabatt auf alle Xelia-Produkte und Updates ab dem Zeitpunkt der Vormerkung!

☐ Hiermit lasse ich mich für die nächste verfügbare Version von Xelia vormerken. Als vorgemerkter Kunde erhalte ich ab dem Zeitpunkt meiner Vormerkung die nächste Version und alle folgenden Updates -wie oben angegeben- mit einem Rabatt von 15 % auf alle Preise. Das gilt auch für Produkterweiterungen und Xelia-Anwendungen, neue Treiber sowie sämtliche Online-Dienste, sofern sie im Zusammenhang mit Xelia angeboten werden (XeliaNetwork). Ich bin mit meiner Vormerkung nicht zur Bestellung oder Abnahme verpflichtet.

X

Datum, Unterschrift

Name, Vorname, Titel

Firma, Abteilung

Straße, Hausnummer

Land, PLZ, Ort

Telefon, Telefax

eMail

Bitte mit
DM 1,00
freimachen

C.-E. Fischer Buchverlag
z.Hd. Herrn Ropielewski
Wegländerstr. 24

70563 Stuttgart

Name, Vorname

Firma, Abteilung

Straße, Hausnummer

Land, PLZ, Ort,

Telefon, Telefax

eMail

Bitte mit
DM 1,00
freimachen

C.-E. Fischer Buchverlag
z.Hd. Herrn Ropielewski
Wegländerstr. 24

70563 Stuttgart

Stichwortverzeichnis

A

Accelerator-Karte: Graphikadapter mit einem Prozessor, der graphische Grundfunktionen ausführt, etwa das Zeichnen von Linien, Rechtecken, Kreisen oder das Füllen von Flächen auf dem Bildschirm. Im Standard-VGA-Modus übernimmt die CPU diese Aufgabe. Graphikkarten mit einem Acceleratorchip beschleunigen daher das System, da elementare Zeichenfunktionen nicht mehr von der CPU ausgeführt werden müssen. Heutige Acceleratorkarten übernehmen auch andere Zeichenaufgaben, etwa das Rendern oder die dreidimensionale Darstellung graphischer Körper.

APAR: Abkürzung für *Authorized Program Analysis Report.* Eine Aufstellung aller Bugs in einem Produkt und Hinweise darauf, durch welches Update bzw. Fix sie beseitigt werden.

Anonymous-FTP: Bezeichnung für den öffentlichen Zugriff auf einen FTP-Server. Für gewöhnlich können nur Benutzer auf einen FTP-Server zugreifen, die über ein entsprechendes Login und Paßwort verfügen. Um den Zugriff auf öffentliche Dateibereiche des Servers zu gewährleisten, richtet man einen Zugang ein, bei dem das Login anonymous heißt und das Paßwort die e-Mail-Adresse des Benutzers ist.

Archie: Ein Informationssystem mit dem nach Dateien auf verschiedenen FTP-Servern gesucht werden kann. Archie verliert wegen der immer mehr steigenden Popularität des WWW an Bedeutung.

ARP: Abkürzung für *Address Resolution Protocol.* Protokoll der TCP/IP-Protokollfamilie, das zur Bestimmung der Hardwareadresse eines Hosts aus dessen IP-Adresse dient.

ASPI: s. Bd. 3

B

Backout: Bezeichnung eines Prozeß, der den früheren Servicelevel eines Produktes, z.B. des Betriebssystems, mit den bei der Installation eines CSD gesicherten Dateien wiederherstellt.

BIOS: Abkürzung für *Basic Input Output System.* Ein Baustein auf der Hauptplatine eines Rechners, der Grundfunktionen zum Zugriff des Prozessors auf Arbeits- und Festspeicher sowie Tastatur, Schnittstellen und Bildschirm bietet. Das BIOS ist die elementarste Schnittstelle, die von Programmen zur Steuerung des Rechners genutzt werden kann. Im Gegensatz zu DOS kann unter OS/2 kein Anwendungsprogramm BIOS-Funktionen aufrufen.

Broadcast: Ein System, das Datenpakete oder Nachrichten an alle Arbeitsstationen eines Netzwerkes schickt. Dazu dient eine dem Netzwerk entsprechende Boradcastadresse.

C

CD: Abkürzung für Compact Disc. Ein optischer Datenträger mit einer Kapazität von ca. 650 MByte. Eine CD ist etwa 1,2 mm dick und besitzt einen Durchmesser von 12 cm. CDs werden gepreßt. Die beschreibbare bzw. wiederbeschreibbare Form der CD wird mit einem Laser beschrieben. CDs dienen der Speicherung von Daten aller Art. Eine CD ist ein Read Only-Medium, d.h. ein sich nach der Herstellung anschließendes Beschreiben ist nicht möglich.

CD-Image: Ein für die Übernahme auf eine CD gefertigtes Abbild in einem speziellen Format einer Ansammlung von Verzeichnissen und Dateien. Das Image beinhaltet den gesamten Inhalt der CD, d.h. alle Dateien sind bereits im richtigen Format mit den dazugehörigen Verzeichnisinformationen

gespeichert. Images sind Quasi-CDs auf der Festplatte und müssen nur noch auf den Zieldatenträger übertragen werden.

CD-Recorder: Ein Laufwerk, daß CD-Rs oder auch CD-RWs beschreiben kann. Das Beschreiben der CD-Rs/CD-RWs erfolgt mit einem Laser, dessen Energie höher ist als zum Abtasten auf CDs gespeicherter Infomationen.

CD-R: Abkürzung für *CD-Recordable*. Eine CD-R stellt eine leere CD dar, die im Gegensatz zu gewöhnlichen CDs keine metallische Reflexionsschicht, sondern eine chemische besitzt, die durch Wärmeeinwirkung (also durch den Laser des CD-Recorders) einmalig verändert werden kann, um Daten aufzunehmen.

CD-RW: Abkürzung für *CD-Rewritable.* Wie die CD-R, nur daß der Datenträger mehrmals beschrieben werden kann. Zur Verarbeitung einer CD-RW ist ein spezieller CD-RW-fähiger CD-Recorder nötig.

CRC-Prüfung: Abkürzung für *Cyclic Redundancy Checksum*. Eine häufig verwandtes Verfahren zur Fehlerbestimmung während Datenübertragungen.

CSD: Abkürzung für *Corrective Service Disk*. Eine Zusammenstellung aktualisierter Dateien für ein Produkt, welche die Erhöhung des Servicelevels dieses Produktes bedingt.

D

Datagramm: Bezeichnung für ein Datenpaket, das alle für eine paketorientierte Datenübermittlung notwendigen Informationen enthält. Unter TCP/IP besteht ein Datagramm aus einem Datenteil und einem Header, der sich aus der Adresse des verschickenden und des empfangenden Hosts zusammensetzt.

Default-Router: Ein Host in einem Netzwerk, an dem alle Daten geschickt werden, die an eine Arbeitsstation adressiert sind, welche sich nicht im eigenen Netzwerk befindet.

DNS: Abkürzung für *Domain Name Server.* Ein Host in einem TCP/IP-Netzwerk, der eine Tabelle aller im Netz vorhandenen Hosts zentral verwaltet.

Domain: Die symbolische und/ oder numerische Bezeichnung einer Netzwerkadresse, die für alle Arbeitsstationen desselben Netzes identisch ist.

F

FAT: s. Bd. 3

FDISK: Dientprogramm zur partitionierung von Festplatten. Unter OS/2 existiert eine Textversion und eine graphische Variante des Programms.

Firewall: Ein als Sicherheitssystem arbeitendes Gateway. Ein als Firewall konfigurierter Rechner erlaubt nur Zugriff autorisierter Hosts und Protokolle auf durch den Firewallrechner geschützte Netzwerkbereiche.

FTP: Abkürzung für *File Transfer Protocol.* Ein Protokoll der TCP/IP-Protokollfamilie, das die Übertragung von Dateien von einem Host zum anderen festlegt.

G

Gateway: Ein Rechner in einem Netzwerk, der eine Verbindung zu anderen Netzwerken herstellt und Daten des einen Netzwerkes in das andere und umgekehrt übermittelt. Gateways können mit unterschiedlichen Sicherheitsfunktionen ausgestattet sein, um den Zugriff auf ein Netzwerk bzw. auf bestimmte Bereiche dieses Netzwerkes zu schützen.

Gopher: Ein auf TCP/IP basierendes Protokoll und ein entsprechendes

graphisches Programm, das einen menüorientierten Zugriff auf Verzeichnisse und Daten anderer Hosts ermöglicht. Durch die immer wachsende Bedeutung des WWW verliert Gopher zugunsten von HTTP an Bedeutung.

Grundebene: Eine Ebene der datentragenden Schicht einer CD, in welche Vertiefungen eingebrannt oder gepreßt werden. Die Grundebene stellt damit einen der beiden definiten Zustände dar, welche für die Speicherung digitalisierter Daten nötig sind. Gegensatz: Pit.

H

Host: Ein Rechner in einem TCP/IP-Netzwerk. Da TCP/IP ein Peer-to-Peer-Netzwerk ist, wird zwischen Client und Server hinsichtlich der einzelnen Arbeitsstationen nicht unterschieden. Jede Station in einem Netz ist vom Prinzip her gleichberechtigt und wird daher als Host (Gastgeber) bezeichnet.

HPFS: s. Bd. 3

HTTP: Abkürzung für *Hypertext Transfer Protocol*. Ein Protokoll der TCP/IP-Protokollfamilie zur Übertragung HTML-kodierter Dokumente.

I

IDE: s. Bd. 3

IP: Abkürzung für *Internet Protocol*. Das Basisprotokoll der TCP/IP-Protokollfamilie, das die Art und Weise einer paketorientierten Datenübertragung vorschreibt.

IP-Adresse: Eine aus vier Oktetten bestehende 32 Bit-Adresse eines Hosts in einem TCP/IP-Netz, welche sowohl den Host selbst auch auch das Netzwerk beschreibt, an das der Host angeschlossen ist. In Dezimalschreibweise wird die IP-Adresse durch vier mit drei Punkten voneinander getrennten Ziffengruppen dargestellt.

ISA-Bus: s. Bd. 3

ISO: Abkürzung für *International Standards Organization*. Gremium zur Standadisierung und Schaffung von Normen, ähnlich dem DIN (Deutsches Institut für Normung).

J

Jumper: Steckbrücke, einfachste Form eines Schalters. Ein Jumper besteht in der Regel aus zwei stiftförmigen Kontakten, welche durch einen Kontakt verbunden oder voneinander getrennt werden können. Jumper werden auf vielen Adapterkarten und auch für Laufwerke zur Konfiguration dieser Komponenten verwendet.

L

Lead-In: Bereich auf einer CD, welcher den Beginn einer Session beschriebt.

Lead-Out: Bereich auf einer CD, welcher das Ende einer Session beschreibt. Der Lead-Out enthält Informationen über die gesamte Session, v.a. ein Verzeichnis aller Dateien, die auf den Datenträger übernommen worden sind.

LIBPATH: Systemsuchpfad für DLLs. Wenn unter OS/2 eine DLL geladen wird, durchsucht das System zunächst die im LIBPATH angegebenen Pfade nach der Bibliothek. Programme, deren DLLs im LIBPATH eingetragen sind, müssen sich nicht um das Laden ihrer DLLs und das Bestimmen der Adressen der Funktionen in diesen DLLs kümmern, da das System diese Arbeit übernimmt. Nachteilig wirkt sich aus, daß das Programm nicht gestartet werden kann, wenn die DLL im vom System durch den LIBPATH ermittelten Pfad nicht auffindbar ist oder wenn der

LIBPATH-Eintrag nicht mehr aktuell ist.

Loader: Bestandteil des Betriebssystems, der Basistreiber und Einheitenmanager sowie den Kernel des Systems beim Starten des Computers in den Arbeitsspeicher lädt. Unter OS/2 befindet sich der Loader in der Datei *os2ldr*.

M

MBR: Abkürzung für *Master Boot Record*. Sektor auf der Festplatte, von dem das Sytem starten kann. Der MBR enthält Informationen für das System, wo der Loader zu finden ist, der den weiteren Startprozeß leitet.

Memory-Files: Im Arbeitsspeicher des Systems angelegte, also virtuelle Dateien. Auf Memory-Files kann man genau so zugreifen wie auf Dateien, die sich auf der Festplatte eines Rechners befinden, wobei der Zugriff allerdings schneller verläuft, da die Datei im Arbeitsspeicher simuliert wird.

Micro Channel Architecture: s. Bd. 3

MIDI-Schnittstelle: s. Bd. 3

MIME: Abkürzung für *Multipurpose Internet Mail Extensions*. Protokoll, das die Übertragung von Daten, die in einem Binärformat (also nicht als Standard ASCII-Text) vorliegen, wie Graphiken, AVI-Files, EXE-Dateien usw., in einer e-Mail erlaubt.

Multisession-Technologie: Technologie, die es gestattet, mehr als nur eine Session auf einer CD/ CD-R/ CD-RW zu speichern.

Multitasking: s. Bd. 3

N

NLS: s. Bd. 3

NFS: Abkürzung für *Network File System*. Installierbares Dateisystem zur Abbildung von Hosts im Netzwerk auf virtuelle Laufwerke der lokalen Maschine.

NNTP: Abkürzung für *Network News Transfer Protocol*. Ein Protokoll der TCP/IP-Protokollfamilie, welches die Übertragung von Nachrichten zwischen News-Servern gestattet.

O

Oktett: Bezeichnung für eine Folge von 8 Bits. Ein Oktett ist damit ein Byte. Bei der Arbeit mit Netzwerken ist der Begriff Oktett statt Byte gebräuchlich, da es Stationen und darauf basierende Systeme gibt, deren Bytes nicht aus acht Bit bestehen. Für uns sind derartige Systeme jedoch irrelevant.

P

PATH: Systemsuchpfad für ausführbare Programmdateien. Beim Start eines Programms durchsucht das System diesen Suchpfad nach der im Startbefehl genannten Programmdatei und kann das Programm laden, ohne daß zuvor in das Programmverzeichnis gewechselt werden müßte.

Partition: s. Bd. 3

Partitionstabelle: s. Bd. 3

Ping: Protokoll aus der TCP/IP-Protokollfamilie zur Überprüfung der Erreichbarkeit eines Hosts in einem TCP/IP-Netzwerk.

Pit: Vertiefung in der Grundebene einer CD zur Darstellung der zweiten Komponente eines Binärcodes zur Speicherung von Daten in einem digitalen Format. Diese Vertiefungen haben ein anderes Reflexionsverhalten als die Bereiche der Grundebene. Beim Gleiten des Laserstrahls über die CD-Oberfläche können die unterschiedlichen Lichtintensitäten nach der Reflexion an den Bereichen der Grundebene und den Pits ermittelt werden, wodurch zwischen zwei Zuständen unterschieden werden kann, und die

Speicherung eines Binärcodes auf dem Datenträger möglich wird.

PM: s. Bd. 1

PPP: s. Bd. 3

Pregab: Bereich zwischen zwei Tracks auf einer CD, der keine Daten enthält. Auf Audio-CDs entspricht der Pregab einer Pause zwischen den einzelnen Musikstücken. Grundsätzlich befindet sich zwischen jedem Track, gleich ob Daten- oder Audiotrack, ein Pregab. Die Länge dieses Bereiches läßt sich mit der CD-Writer Software festlegen.

R

Repeater: s. Bd. 3

RFC: Abkürzung für *Request For Comments.* Die RFCs sind Normen gleich, welche die verschiedenen TCP/IP-Protokolle festlegen und beschreiben.

Router: Eine Station, die mehrere Netzwerke miteinander verbindet, um eine Kommunikation zwischen den Netzen zu ermöglichen. Ein Router braucht demnach mindestens zwei Interfaces und unter TCP/IP zwei unterschiedliche Netzwerkadressen.

S

SCSI: s. Bd. 3

Sektoren: Bereich auf einem Datenträger, der Nutzdaten enthält. Sektoren können in größeren logische Gruppen zusammengefaßt sein, z.B. im Falle der CD in einem Track.

Session: *1.* Sitzung, in der ein Programm läuft. *2.* Bereich auf einer CD, der TOC, Lead-In, Lead-Out und Nutzdaten enthält.

SLIP: s. Bd. 3

SMTP: s. Bd. 3

T

TCP: Abkürzung für *Transmission Control Protocol.* Protokoll der TCP/IP-Protokollfamilie, welches eine gesicherte Datenübertragung auf IP-Basis ermöglicht. TCP überwacht Datenübertragungen, führt Fehlerkorrekturen durch und setzt die von IP gesendeten Datenpakete wieder zur ursprünglichen Nachricht zusammen bzw. zerlegt eine Nachricht in Datenpakete zum Versand mit IP.

Telnet: Bezeichnung für *Remote Terminal Protocol.* Ein Protokoll der TCP/IP-Protokollfamilie, das den Aufbau einer Verbindung zu einem Host im gleichen oder in einem fremden Netzwerk über eine Sitzung im Textmodus aufbaut. Über eine Telnet-Sitzung können Programme auf einem entfernten Rechner von der lokalen Arbeitsstation aus gestartet und bedient werden.

Track: Bereich auf einer CD, der für die Aufnahme der Nutzdaten zuständig ist. Es gibt unterschiedliche Arten von Tracks (Daten- und Audiotracks). Tracks sind unterteilt in Sektoren.

Trap: s. Bd. 3

U

UDP: Abkürzung für *User Datagram Protocol.* Protokoll der TCP/IP-Protokollfamilie, das die paketorientierte Datenübertragung an bestimmte Programme oder Benutzer eines Hosts ermöglicht. UDP ist wie IP ein ungesichertes Protokoll.

V

VLB: s. Bd. 3

W

Wartungspartition: Partition auf der Festplatte eines Systems, die eine minimierte Variante des Betriebssystems zum Systemstart und für Wartungszwecke enthält.

WsoD: s. Bd. 3

Software

Hobbes Archiv
http://hobbes.nmsu.edu/pub/os2

LEO Archiv
http://www.leo.org/archiv/software/os2

RSJ Homepage
http://www.rsj.de

OS/2 Shareware
http://www.bmtmicro.com

IBM Visual Age for Java
http://www.service.software.ibm.com/dl/vajava/vjavadl-d

OS/2 Shareware BBS Website
http://www.os2bbs.com

Papyrus Homepage
http://www.rom-lgic.de

PMView Homepage
http://www.pmview.com

IRC-Software Open Chat
http://www.escape.com/~mikh/openchat/

SciTech Homepage
http://www.scitechsoft.com

PPP-Software/ Dialer Injoy/2
http://www.fx.dk/injoy

ProNews Homepage
http://www.program.com/panacea/pronews/download.shtml

Virtual Pascal
http://www.fprint.co.uk/vpascal.html

Fixes und Updates

IBM Techn. Außendienst Mainz
http://servicepac.mainz.ibm.de

Fixpacks über FTP
ftp.software.ibm.com/ps/products/os2/fixes

Informationen

FAQ-Liste zu OS/2 Peer
http://stpc6.khi.de/team/faq/os2peer.htm

IBM Alphaworks - Informationen zu Java
http://alphaworks.ibm.com/formula

Benutzergruppen

OS/2 User Group Belgien
http://www.os2ugbe.org

OS/2 User Association Schweiz
http://www.os2web.inserto.ch/os2ua/index.html

Händler

OS/2 Hard- und Software
AP-COM (Andreas Pieter Computer)
http://www.ap-com.de

OS/2 Software
Softwareshop Deckarm & Co.
http://www.softwareshop.com

Verlag

Homepage
http://www.cefischer.de

OS/2 Only!
http://www.cefischer.de/deutsch/os2/os2only/only.htm

Xelia
http://www.cefischer.de/deutsch/os2/xelia/xelia.htm

Die OS/2 Only! ist das anwenderorienterte OS/2-Magazin nur für OS/2 mit regelmäßigen Vorstellungen von OS/2-Software und Tests voll OS/2-tauglicher Hardware, ausführlichen Programmier- und Anwendungsworkshops, Treiber- und Fixpackunterstützung.

Das Magazin stellt Marktneuheiten vor, bietet Produktübersichten und gibt wichtige Tips & Tricks, die vor allem dem Endanwender nützen. Es bietet Hilfestellung bei Problemen, hilft beim Umsteigen auf das Betriebssystem OS/2 von anderen Plattformen und vermittelt in umfangreichen Artikeln und Workshops alles Nötige zum Programmieren in allen wichtigen Sprachen und zur Konfiguration von Netzwerken und Einzelplatzsystemen.

Außerdem in jeder Ausgabe: Ein umfangreiches Stichwortverzeichnis zum schnellen Auffinden von Informationen und ein Lexikon zur Erklärung aller wichtigen Begriffe.

CEF C.-E. Fischer
Buchverlag, Stuttgart

DM 24,80 Inland
Euro 13,70 Ausland
unverb. Preisempfehlung

ISSN 1436-8307

OS/2 Only! Bd. 2 - Ausgabe 2/99